

# 3

## **Uso de Funciones de Una Sola Fila para Personalizar la Salida**

ORACLE®

Copyright © 2004, Oracle. Todos los derechos reservados.

# Objetivos

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

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

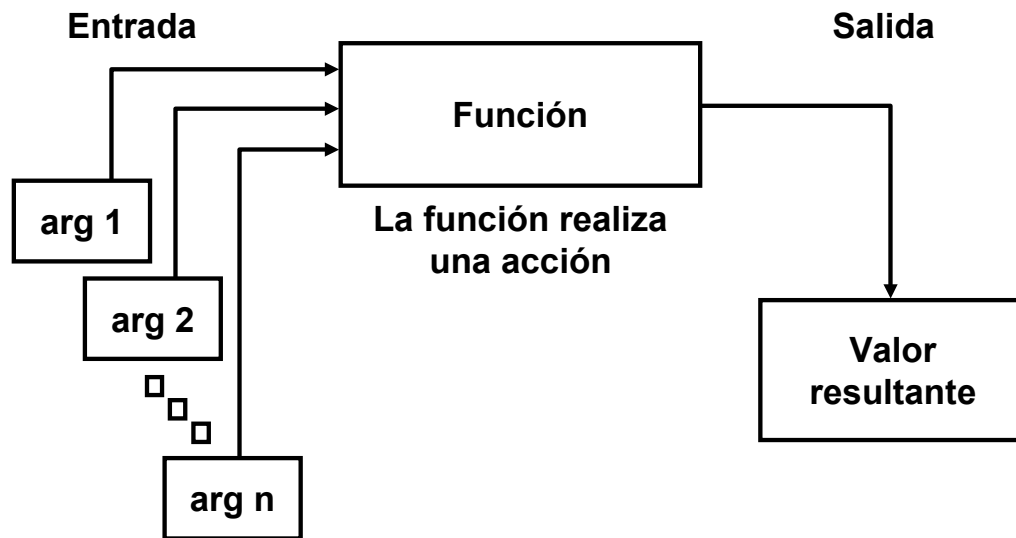
ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Objetivos

Las funciones hacen que el bloque de consulta sea más potente y se utilizan para manipular valores de datos. Ésta es la primera de dos lecciones que examinan las funciones. Se centra en las funciones de una sola fila de carácter, numéricas y de fecha, así como en las que convierten datos de tipo a otro (por ejemplo, conversión de datos de carácter a datos numéricos).

# Funciones SQL



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones SQL

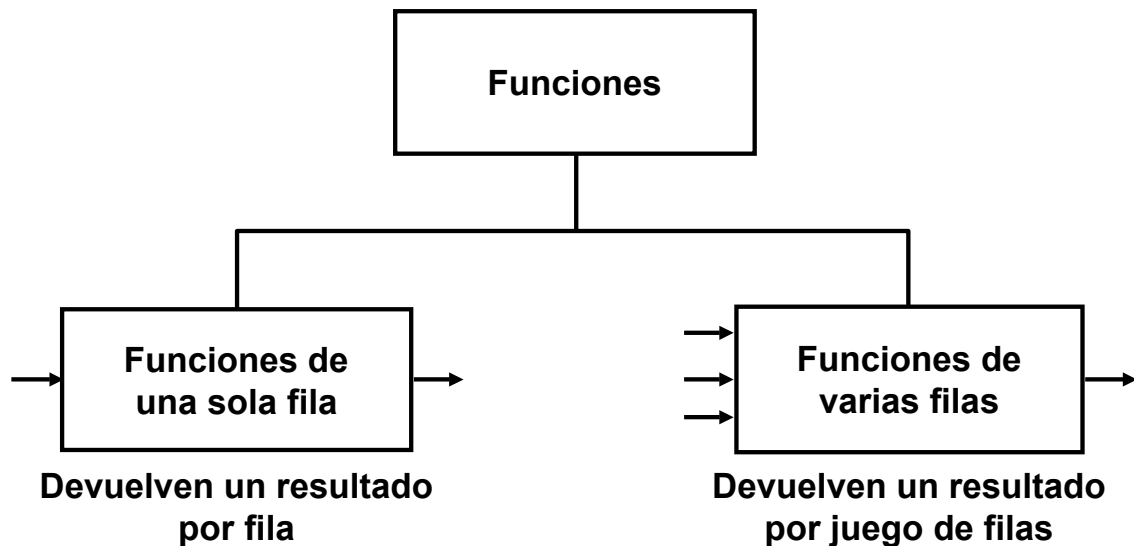
Las funciones son una característica muy potente de SQL. Se pueden utilizar para:

- Realizar cálculos en datos
- Modificar elementos de datos individuales
- Manipular la salida para grupos de filas
- Formatear fechas y números para su visualización
- Convertir tipos de datos de columnas

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

**Nota:** En su mayor parte, las funciones que se describen en esta lección son específicas de la versión Oracle de SQL.

# Dos Tipos de Funciones SQL



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones SQL (continuación)

Hay dos tipos de funciones:

- Funciones de una sola fila
- Funciones de varias filas

### Funciones de una Sola Fila

Estas funciones operan sólo en filas únicas y devuelven un resultado por fila. Hay diferentes tipos de funciones de una sola fila. Esta lección trata las siguientes:

- De carácter
- Numéricas
- De fecha
- De conversión
- Generales

### Funciones de Varias Filas

Las funciones pueden manipular grupos de filas para dar un resultado por grupo de filas. Estas funciones se conocen también como *funciones de grupo* (y se describen en una lección posterior).

**Nota:** Para obtener más información y una lista completa de las funciones disponibles y su sintaxis, consulte *Oracle SQL Reference*.

# Funciones de una Sola Fila

## Las funciones de una sola fila:

- **Manipulan elementos de datos**
- **Aceptan argumentos y devuelven un valor**
- **Actúan en cada fila que se devuelve**
- **Devuelven un resultado por fila**
- **Pueden modificar el tipo de datos**
- **Se pueden anidar**
- **Aceptan argumentos que pueden ser una columna o una expresión**

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

ORACLE

Copyright © 2004, Oracle. 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 más argumentos y devuelven un valor para cada fila que devuelve la consulta. Un argumento puede ser uno de los siguientes:

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

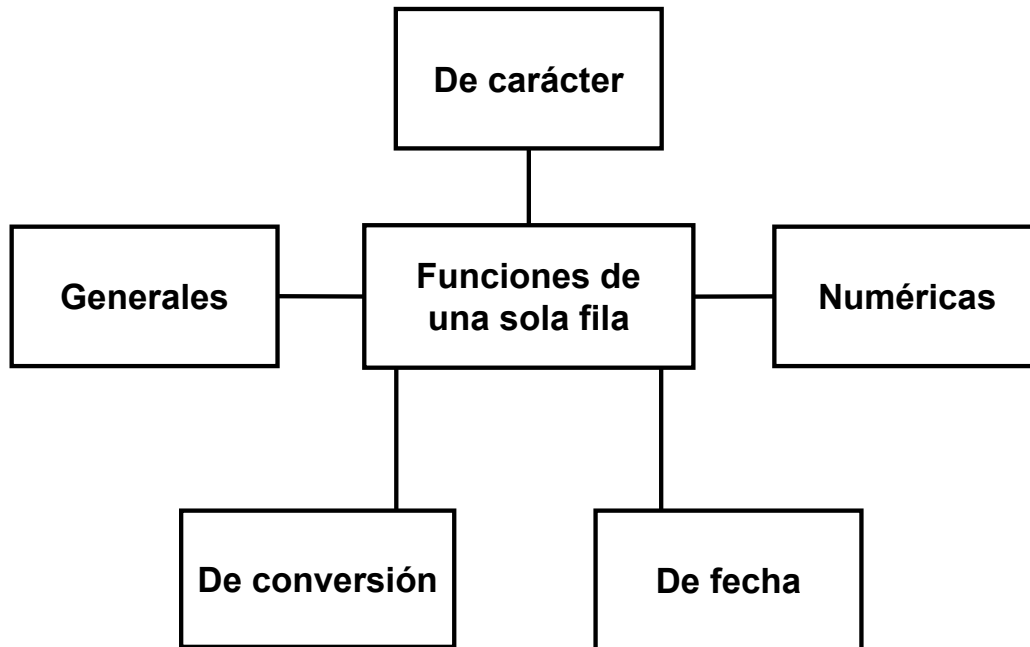
Las características de las funciones de una sola fila son:

- Actúan en cada fila que se devuelve en la consulta
- Devuelven un resultado por fila
- Posiblemente devuelven un valor de datos de un tipo diferente a aquel al que se hace referencia
- Posiblemente esperan uno o más argumentos
- Se pueden utilizar en cláusulas SELECT, WHERE y ORDER BY; se pueden anidar

En la sintaxis:

<i>function_name</i>	es el nombre de la función
<i>arg1, arg2</i>	es cualquier argumento que vaya a utilizar la función. Se puede representar con un nombre de columna o una expresión.

# Funciones de una Sola Fila



ORACLE

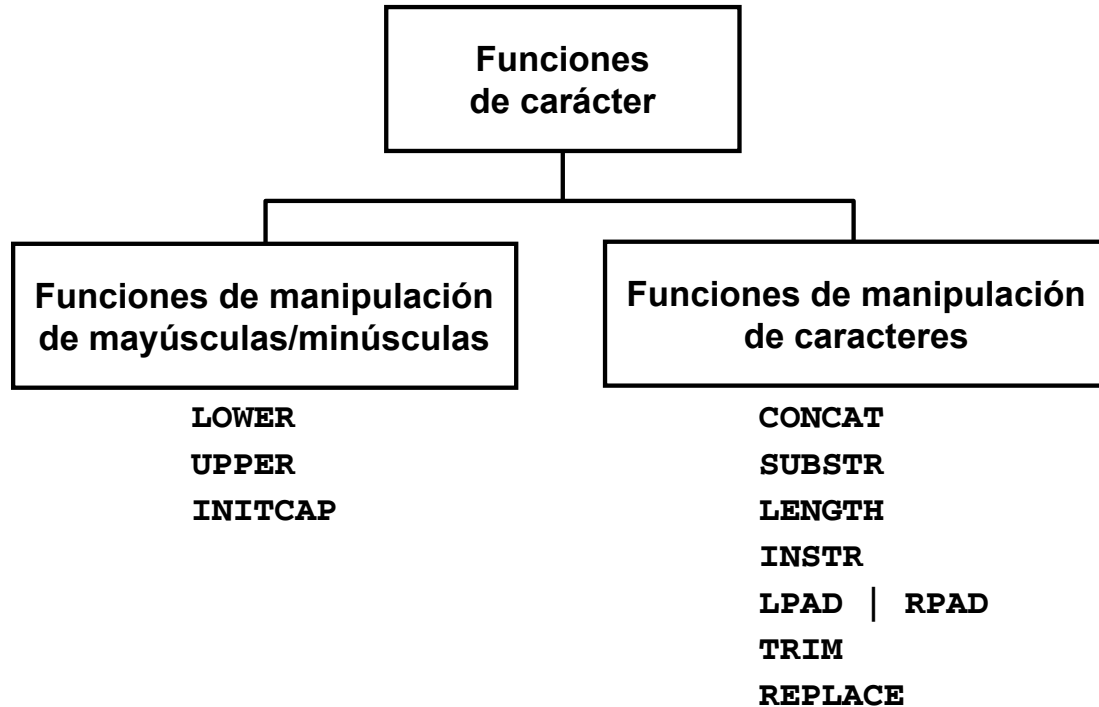
Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones de una Sola Fila (continuación)

Esta lección trata estas funciones de una sola fila:

- **Funciones de carácter:** Aceptan la entrada de caracteres y pueden devolver valores de carácter y numéricos
- **Funciones numéricas:** Aceptan la entrada de números y devuelven valores numéricos
- **Funciones de fecha:** Operan en valores del tipo de datos DATE (todas las funciones de fecha devuelven un valor del tipo de datos DATE excepto la función MONTHS\_BETWEEN, que devuelve un número.)
- **Funciones de conversión:** Convierten un valor de un tipo de datos en otro.
- **Funciones Generales:**
  - NVL
  - NVL2
  - NULLIF
  - COALESCE
  - CASE
  - DECODE

# Funciones de Carácter



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones de Carácter

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

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

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

**Nota:** Las funciones analizadas en esta lección únicamente suponen una parte de las disponibles.

## Funciones de Carácter (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 una posición <i>m</i> para iniciar la búsqueda y la incidencia <i>n</i> de la cadena. <i>m</i> y <i>n</i> tienen por defecto el valor 1, lo que significa iniciar la búsqueda al principio de la búsqueda e informar de la primera incidencia.
<code>LPAD(column expression, n, 'string')</code> <code>RPAD(column expression, n, 'string')</code>	Rellena el valor de carácter justificado a la derecha hasta un ancho total de <i>n</i> posiciones de carácter. Rellena el valor de carácter justificado a la izquierda hasta un ancho total de <i>n</i> posiciones de carácter.
<code>TRIM(leading trailing both, trim_character FROM trim_source)</code>	Le permite recortar caracteres de cabecera o finales (o ambos) de una cadena de caracteres. Si <i>trim_character</i> o <i>trim_source</i> son literales de carácter, los debe poner entre comillas simples. Esto está disponible en Oracle8i y versiones posteriores.
<code>REPLACE(text, search_string, replacement_string)</code>	Busca en una expresión de texto una cadena de caracteres y, de encontrarla, la sustituye por una cadena de sustitución especificada



# Funciones de Manipulación de Mayúsculas/Minúsculas

Estas funciones convierten en mayúsculas y minúsculas las cadenas de caracteres:

Función	Resultado
LOWER('SQL Course')	sql course
UPPER('SQL Course')	SQL COURSE
INITCAP('SQL Course')	Sql Course

ORACLE

Copyright © 2004, Oracle. 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 con mayúsculas/minúsculas mezcladas o mayúsculas a minúsculas
- **UPPER:** Convierte cadenas de caracteres con mayúsculas/minúsculas mezcladas o minúsculas a mayúsculas
- **INITCAP:** Convierte la primera letra de cada palabra a mayúsculas y el resto de las 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, nombre y número de departamento del 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de Funciones de Manipulación de Mayúsculas/Minúsculas

La diapositiva muestra el número de empleado, nombre y 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 y minúsculas adecuadas, el nombre higgins no encuentra correspondencia en la tabla y no se selecciona ninguna fila.

La cláusula WHERE de la segunda sentencia SQL especifica que el nombre de empleado de la tabla EMPLOYEES se compare con higgins, con lo que convierte la columna LAST\_NAME a minúsculas por motivos de comparación. Como ambos nombres están ahora en minúsculas, se encuentra una correspondencia y se selecciona una fila. La cláusula WHERE se puede reescribir del modo siguiente para crear el mismo resultado:

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

El nombre de la salida aparece como se almacenó en la base de datos. Para mostrar el nombre sólo con la primera letra 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*****
REPLACE ('JACK and JUE','J','BL')	BLACK and BLUE
TRIM('H' FROM 'HelloWorld')	elloWorld

ORACLE

Copyright © 2004, Oracle. 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 (el uso de parámetros con CONCAT se limita a dos.)
- **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 carácter justificado a la derecha
- **RPAD:** Rellena el valor de carácter 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 caracteres, debe ponerlo entre comillas simples.)

**Nota:** Puede utilizar funciones como UPPER y LOWER con la sustitución ampersand. Por ejemplo, utilice UPPER(' &job\_title') para que el usuario no tenga que introducir el cargo en un modo específico de mayúsculas/minúsculas.

# Uso de Funciones de Manipulación de Caracteres

```

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

1

2

3

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de Funciones de Manipulación de Caracteres

El ejemplo de la diapositiva muestra los nombres y los apellidos de los empleados unidos, la longitud del apellido de empleado y la posición numérica de la letra *a* en el apellido del empleado para todos los empleados que incluyan la cadena REP en la cuarta posición del identificador de puesto.

### Ejemplo

Modifique la sentencia SQL de la diapositiva para que muestre los datos para los empleados cuyos apellidos terminen 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 al decimal especificado
- **TRUNC:** Trunca el valor al decimal especificado
- **MOD:** Devuelve el resto de la división

Función	Resultado
<b>ROUND (45.926, 2)</b>	<b>45.93</b>
<b>TRUNC (45.926, 2)</b>	<b>45.92</b>
<b>MOD (1600, 300)</b>	<b>100</b>

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones Numéricas

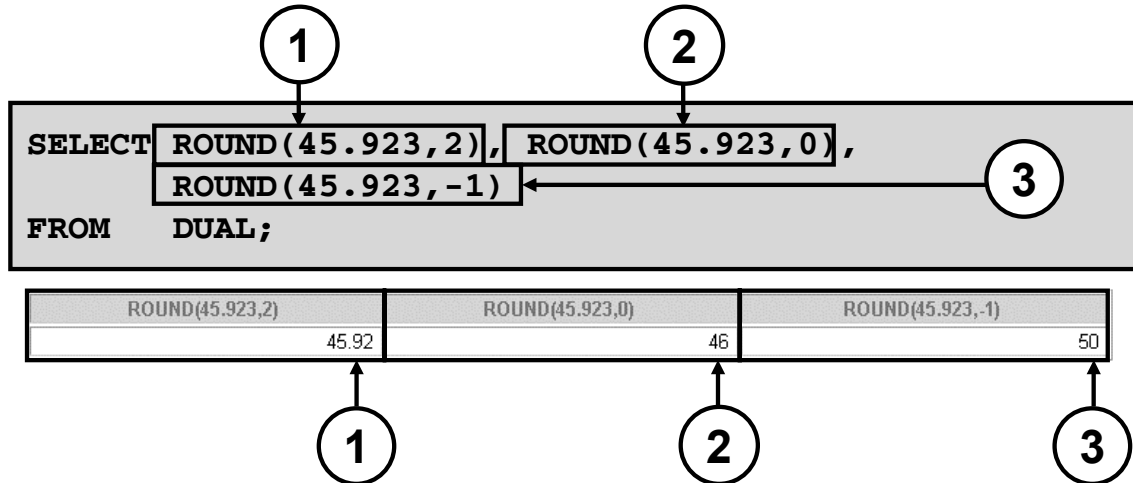
Las funciones numéricas aceptan la entrada de números y devuelven valores numéricos. Esta sección describe algunas de las funciones numéricas.

Función	Objetivo
<code>ROUND (column   expression, n)</code>	Redondea la columna, la expresión o el valor a <i>n</i> posiciones decimales o, si se omite <i>n</i> , a ninguna (Si <i>n</i> es negativo, se redondean los números a la izquierda del separador decimal.).)
<code>TRUNC (column   expression, n)</code>	Trunca la columna, la expresión o el valor a <i>n</i> posiciones decimales o, si se omite <i>n</i> , opta por el valor por defecto de cero
<code>MOD (m, n)</code>	Devuelve el resto de <i>m</i> dividido por <i>n</i>

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

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

## Uso de la Función ROUND



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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

### Función ROUND

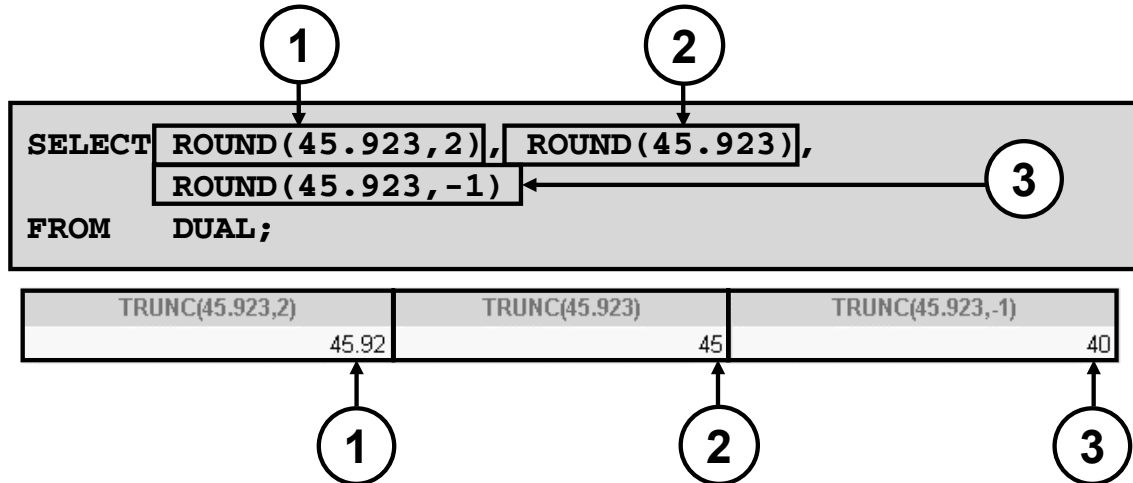
La función ROUND redondea la columna, la expresión o el valor a *n* posiciones decimales. Si el segundo argumento es 0 o falta, el valor se redondea a cero posiciones decimales. Si el segundo argumento es 2, el valor se redondea a dos posiciones decimales. A la inversa, si el segundo argumento es -2, el valor se redondea a dos posiciones decimales a la izquierda (redondeando a la unidad más cercana a 10).

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

### Tabla DUAL

La tabla DUAL es propiedad del usuario SYS y pueden acceder a ella todos los usuarios. Contiene una columna, DUMMY, y una fila con el valor X. La tabla DUAL resulta útil si desea devolver un valor sólo una vez (por ejemplo, el valor de una constante, una pseudocolumna o una expresión que no se deriva de una tabla con datos de usuario). La tabla DUAL se utiliza generalmente para la integridad de sintaxis de la cláusula SELECT, ya que las sentencias SELECT y FROM son obligatorias y varios cálculos no necesitan seleccionar de tablas reales.

## Uso de la Función TRUNC



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

### Función TRUNC

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

La función TRUNC trabaja con argumentos parecidos a los de la función ROUND. Si el segundo argumento es 0 o falta, el valor se trunca a cero posiciones decimales. Si el segundo argumento 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. Si el segundo argumento es -1, el valor se trunca a una posición decimal a la izquierda.

Como la función ROUND, la función TRUNC también se puede utilizar con funciones de fecha.

## Uso de la Función MOD

Para todos los empleados con cargo de representante de ventas, calcule el resto del salario una vez dividido por 5.000.

```
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

Copyright © 2004, Oracle. Todos los derechos reservados.

### Función MOD

La función MOD busca el resto del primer argumento dividido por el segundo argumento. El ejemplo de la diapositiva calcula el resto del salario después de dividirlo por 5.000 para todos los empleados cuyo identificador de puesto es 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 y segundos.
- El formato de visualización de fecha por defecto es DD-MON-RR.
  - Le permite especificar fechas del siglo XXI en el siglo XX especificando sólo los dos últimos dígitos del año
  - Le permite almacenar fechas del siglo XX en el siglo XXI de la misma manera

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date < '01-FEB-88';
```

LAST_NAME	HIRE_DATE
King	17-JUN-87
Whalen	17-SEP-87

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Formato de Fecha de Oracle

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

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

En el ejemplo de la diapositiva, la salida de la columna HIRE\_DATE se muestra en el formato por defecto DD-MON-RR. Sin embargo, las fechas no se almacenan en la base de datos en este formato. Se almacenan todos los componentes de fecha y hora. Así pues, aunque una fecha HIRE\_DATE como 17-JUN-87 se muestra como día, mes y año, también hay información de *hora* y de *siglo* asociada a la fecha. Los datos completos podrían ser 17 de junio de 1987, 5:10:43 p.m.

## Formato de Fecha de Oracle (continuación)

Esta fecha se almacena internamente así:

SIGLO	AÑO	MES	DÍA	HORA	MINUTO	SEGUNDO
19	87	06	17	17	10	43

### Siglos y el Año 2000

Al insertar un registro con una columna de fecha en una tabla, la información de *siglo* se selecciona en la función `SYSDATE`. Sin embargo, cuando la columna de fecha se muestra en la pantalla, el componente de siglo no se muestra (por defecto).

El tipo de datos `DATE` almacena siempre la información de año como número de cuatro dígitos internamente: dos dígitos para el siglo y dos para el año. Por ejemplo, la base de datos Oracle almacena el año como 1987 ó 2004, y no simplemente como 87 ó 04.

# Trabajo con Fechas

**SYSDATE** es una función que devuelve:

- Fecha
- Hora

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Función **SYSDATE**

**SYSDATE** es una función de fecha que devuelve la fecha y la hora actuales del servidor de bases de datos. Puede utilizar **SYSDATE** igual que cualquier otro nombre de columna. Por ejemplo, puede mostrar la fecha actual seleccionando **SYSDATE** en una tabla. Se suele seleccionar **SYSDATE** en una tabla ficticia denominada **DUAL**.

### Ejemplo

Muestre la fecha actual mediante la tabla **DUAL**.

```
SELECT SYSDATE  
FROM DUAL;
```

SYSDATE
28-SEP-01

# Aritmética de Fechas

- **Sume o reste un número a una fecha para obtener un valor de fecha resultante.**
- **Reste dos fechas para calcular el número de días entre dichas fechas.**
- **Sume horas a una fecha dividiendo el número de horas entre 24.**

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Aritmética de Fechas

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

Puede utilizar 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 a una fecha
fecha – fecha	Número de días:	Resta una fecha a 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Aritmética de Fechas (continuación)

El ejemplo de la diapositiva muestra el apellido y el número de semanas de empleo para todos los empleados 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 un trabajador lleva empleado.

**Nota:** SYSDATE es una función SQL que devuelve la fecha y la hora actuales. Sus resultados pueden diferir de los del ejemplo.

Si se resta una fecha más actual a una más antigua, la diferencia será un número negativo.

# Funciones de Fecha

Función	Resultado
<b>MONTHS_BETWEEN</b>	Número de meses entre dos fechas
<b>ADD_MONTHS</b>	Agrega meses de calendario a una fecha
<b>NEXT_DAY</b>	Día siguiente a la fecha especificada
<b>LAST_DAY</b>	Último día del mes
<b>ROUND</b>	Redondea la fecha
<b>TRUNC</b>	Trunca la fecha

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones de Fecha

Las funciones de fechas operan en fechas de Oracle. Todas las funciones de fecha devuelven un valor del tipo de datos 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*)**: Agrega un número *n* de meses de calendario a *date*. El valor de *n* debe ser un entero y puede ser negativo.
- **NEXT\_DAY(*date*, '*char*')**: Busca la fecha del siguiente día de la semana especificado ( '*char*' ) después de *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 que contiene *date*
- **ROUND(*date*[, '*fmt*'])**: Devuelve *date* redondeado a la unidad especificada por el modelo de formato *fmt*. Si se omite el modelo de formato *fmt*, *date* se redondeará al día más cercano.
- **TRUNC(*date*[, '*fmt*'])**: Devuelve *date* con la porción de tiempo del día truncada a la unidad especificada por el modelo de formato *fmt*. Si se omite el modelo de formato *fmt*, *date* se trunca al día más cercano.

Esta lista es un subconjunto de las funciones de fecha disponibles. Los modelos de formato se describirán posteriormente en esta lección. Ejemplos de modelos de formato son el mes y el año.

# Uso de Funciones de Fecha

Función	Resultado
<b>MONTHS_BETWEEN</b> ( '01-SEP-95' , '11-JAN-94' )	<b>19.6774194</b>
<b>ADD_MONTHS</b> ( '11-JAN-94' , 6 )	<b>'11-JUL-94'</b>
<b>NEXT_DAY</b> ( '01-SEP-95' , 'FRIDAY' )	<b>'08-SEP-95'</b>
<b>LAST_DAY</b> ( '01-FEB-95' )	<b>'28-FEB-95'</b>

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones de Fecha (continuación)

Por ejemplo, muestre el número de empleado, la fecha de contratación, el número de meses de empleo, la fecha de revisión semestral, el primer viernes tras la fecha de contratación y el mes de contratación de todos los empleados que lleven contratados 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

Supongamos que **SYSDATE** = '25-JUL-03':

Función	Resultado
<b>ROUND</b> (SYSDATE, 'MONTH')	01-AUG-03
<b>ROUND</b> (SYSDATE, 'YEAR')	01-JAN-04
<b>TRUNC</b> (SYSDATE, 'MONTH')	01-JUL-03
<b>TRUNC</b> (SYSDATE, 'YEAR')	01-JAN-03

ORACLE

Copyright © 2004, Oracle. 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 tanto, puede redondear fechas al año o al mes más cercanos.

### Ejemplo

Compare las fechas de contratación de todos los empleados que empezaron en 1997. Muestre el número de empleado, la fecha de contratación y el mes de inicio mediante 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: Visión General de la Parte 1

**Esta práctica cubre los temas siguientes:**

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

ORACLE

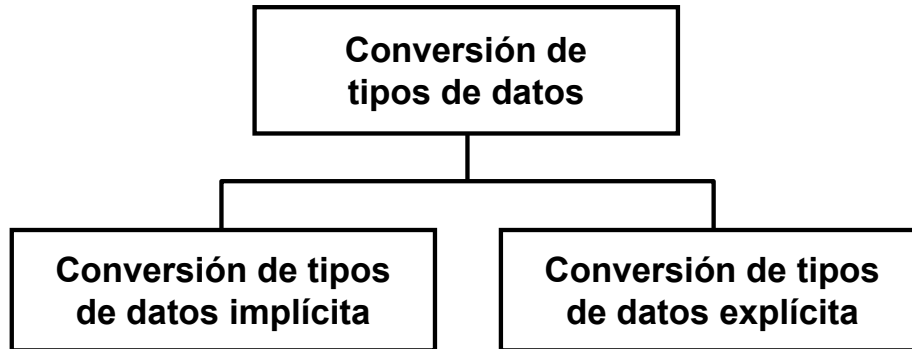
Copyright © 2004, Oracle. Todos los derechos reservados.

### **Práctica 3: Visión General de la Parte 1**

La práctica de la Parte 1 de esta lección proporciona varios ejercicios que utilizan diferentes funciones disponibles para los tipos de datos de carácter, numérico y de fecha.

Para la Parte 1, conteste a las preguntas 1–6 al final de la lección.

# Funciones de Conversión



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones de Conversión

Además de los tipos de datos Oracle, las columnas de tablas de una base de datos Oracle se pueden definir mediante tipos de datos ANSI, DB2 y SQL/DS. Sin embargo Oracle Server convierte internamente esos tipos de datos a tipos de datos Oracle.

En algunos casos, Oracle Server utiliza datos de un tipo de datos donde espera datos de un tipo de datos diferente. Cuando sucede esto, Oracle Server puede convertir automáticamente los datos al tipo de datos esperado. Esta conversión de tipos de datos lo puede realizar *implícitamente* Oracle Server, o *explícitamente* el usuario.

Las conversiones de tipos de datos implícitas funcionan de acuerdo con las reglas que se explican en las dos próximas diapositivas.

Las conversiones de tipos de datos explícitas se realizan mediante las funciones de conversión. Las funciones de conversión convierten un valor de un tipo de dato a otro. Generalmente, la forma de los nombres de función utiliza la convención *data type* TO *data type*. El primer tipo de datos es el de entrada; el segundo, el de salida.

**Nota:** Aunque está disponible la conversión de tipos de datos implícita, se recomienda que realice una conversión de tipos de datos explícita para asegurar la fiabilidad de las sentencias SQL.

# Conversión de Tipos de Datos Implícita

Para las asignaciones, Oracle Server puede convertir automáticamente:

De	A
VARCHAR2 o CHAR	NUMBER
VARCHAR2 o CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Conversión de Tipos de Datos Implícita

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

Por ejemplo, la expresión `hire_date > '01-JAN-90'` da como resultado la conversión implícita de la cadena `'01-JAN-90'` a una fecha.

# Conversión de Tipos de Datos Implícita

**Para la evaluación de expresiones, Oracle Server puede convertir automáticamente:**

De	A
<b>VARCHAR2 O CHAR</b>	<b>NUMBER</b>
<b>VARCHAR2 O CHAR</b>	<b>DATE</b>

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

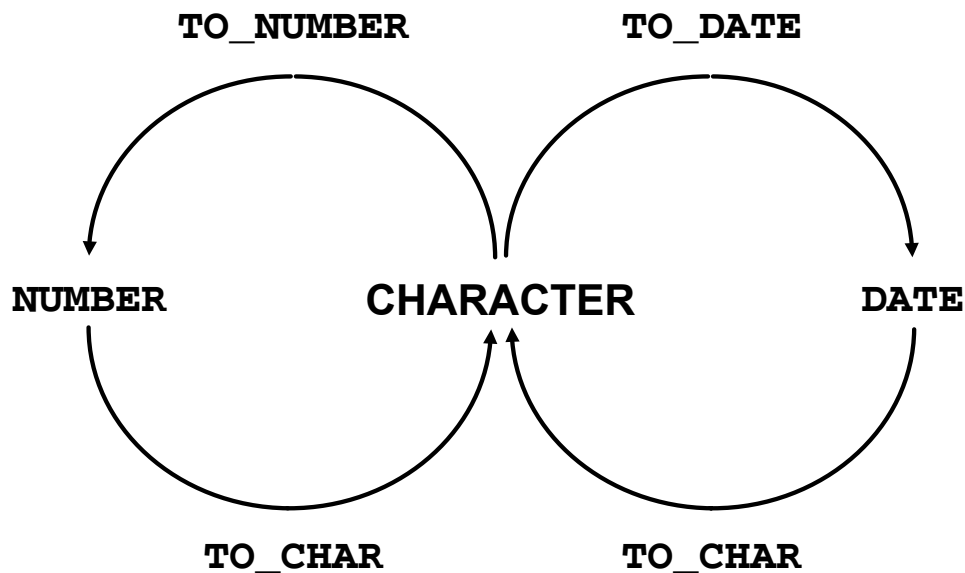
## Conversión de Tipos de Datos Implícita (continuación)

En general, Oracle Server utiliza la regla para expresiones cuando se necesita una conversión de tipo de datos en lugares no cubiertos por una regla para conversiones de asignación.

Por ejemplo, la expresión `salary = '20000'` da como resultado la conversión implícita de la cadena `'20000'` al número 20000.

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

# Conversión de Tipos de Datos Explícita



ORACLE

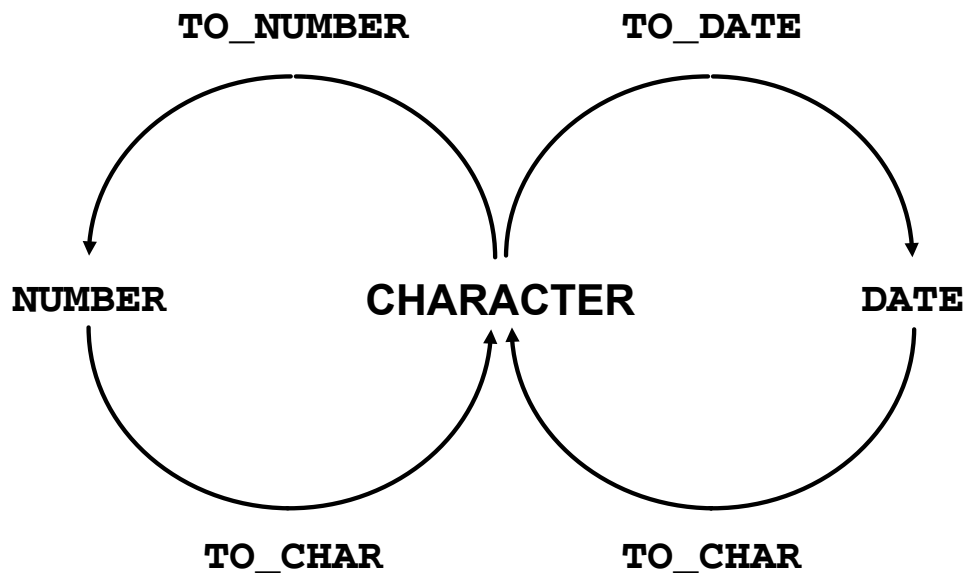
Copyright © 2004, Oracle. Todos los derechos reservados.

## Conversión de Tipos de Datos Explícita

SQL proporciona tres funciones para convertir un valor de un tipo de datos a 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 a 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 son devueltos por elementos de formato numérico:</p> <ul style="list-style-type: none"><li>• Carácter decimal</li><li>• Separador de grupos</li><li>• Símbolo de divisa local</li><li>• Símbolo de divisa internacional</li></ul> <p>Si se omite <i>nlsparms</i> o cualquier otro parámetro, esta función utiliza los valores de parámetros por defecto para la sesión.</p>

# Conversión de Tipos de Datos Explícita



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Conversión de Tipos de Datos Explícita (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 lenguaje en que se devolverán los nombres y las abreviaturas de mes y de día. Si se omite este parámetro, esta función utiliza los lenguajes 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 con el formato especificado por el modelo de formato opcional <i>fmt</i> .  El parámetro <i>nlsparams</i> tiene el mismo objetivo en esta función que en la función <code>TO_CHAR</code> de 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 el <i>fmt</i> que se haya especificado. Si se omite <i>fmt</i> , el formato es DD-MON-YY.  El parámetro <i>nlsparams</i> tiene el mismo objetivo en esta función que en la función <code>TO_CHAR</code> de conversión de fecha.

## Conversión de Tipos de Datos Explícita (continuación)

**Nota:** Las lista de funciones mencionadas en esta lección únicamente suponen una parte de las funciones de conversión disponibles.

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

# Uso de la Función TO\_CHAR con Fechas

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

## El modelo de formato:

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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Visualización de una Fecha en un Formato Específico

Anteriormente, todos los valores de datos 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 debe ir entre comillas simples y es sensible a mayúsculas/minúsculas.
- El modelo de formato puede incluir cualquier elemento de formato de fecha. Asegúrese de separar el valor de fecha del modelo de formato con una coma.
- Los nombres de días y meses de la salida se rellenan automáticamente con espacios en blanco.
- Para eliminar espacios en blanco rellenos o para suprimir ceros iniciales, utilice el elemento *fm* del modo de relleno.
- Puede formatear el campo de caracteres resultante con el comando COLUMN de *iSQL\*Plus* (que se describe 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

Elemento	Resultado
<b>YYYY</b>	Año completo con números
<b>YEAR</b>	Nombre completo de año con letras (en inglés)
<b>MM</b>	Valor de dos dígitos para el mes
<b>MONTH</b>	Nombre completo del mes
<b>MON</b>	Abreviatura de tres letras del mes
<b>DY</b>	Abreviatura de tres letras del día de la semana
<b>DAY</b>	Nombre completo del día de la semana
<b>DD</b>	Día del mes con números

ORACLE®

Copyright © 2004, Oracle. Todos los derechos reservados.

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

Elemento	Descripción
SCC o CC	Siglo; Oracle Server agrega - a las fechas a. C.
Años en fechas YYYY o SYYYY	Año; Oracle Server agrega - a las fechas a. C.
YYY o YY o Y	Los últimos tres, dos o uno dígitos del año
Y,YYY	Año con una coma en esa posición
IYYY, IYY, IY, I	Año de cuatro, tres, dos o un dígitos basado en el estándar ISO
SYEAR o YEAR	Nombre completo de año con letras; Oracle Server agrega - a las fechas a. C.
BC o AD	Indica un año a.C. o d.C.
B.C. o A.D.	Indica un año a.C. o d.C., con puntos
Q	Trimestre
MM	Mes: valor de dos dígitos
MONTH	Nombre del mes relleno con espacios en blanco hasta la longitud de nueve caracteres.
MON	Nombre del mes, abreviatura de tres letras
RM	Mes con números 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 relleno con espacios en blanco hasta la longitud de nueve caracteres.
DY	Nombre del día, abreviatura de tres letras
J	Día juliano; número de días desde el 31 de diciembre de 4713 a.C.

# Elementos del Modelo de Formato de Fecha

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

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

- Agregue cadenas de caracteres poniéndolas entre comillas dobles:

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

- Utilice sufijos para escribir el nombre completo de los números:

ddsptth	fourteenth
---------	------------

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Elemento de Formato de Fecha: Formatos de Hora

Utilice los formatos que se muestran en las tablas siguientes para mostrar información de hora y literales y para cambiar números en cifras a números en letras.

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 desde la medianoche (0–86399)

## Otros Formatos

Elemento	Descripción
/ . ,	La puntuación se reproduce en el resultado.
“de”	Las comillas se reproducen en el resultado.

## Especificación de Sufijos para Influir en la Visualización de Números

Elemento	Descripción
TH	Número ordinal (por ejemplo, DDTH para 4TH)
SP	Número completo con letras (por ejemplo, DDSP para FOUR)
SPTH o THSP	Número ordinal completo con letras (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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función TO\_CHAR con Fechas

La sentencia SQL de la diapositiva 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 diapositiva para mostrar las fechas en un formato que aparezca como “Seventeenth of June 1987 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

...

Observe que el mes sigue el modelo de formato especificado; dicho de otro modo, la primera letra va en mayúsculas y el resto en minúsculas.

## Uso de la Función TO\_CHAR con Números

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

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

Elemento	Resultado
9	Representa un número
0	Muestra ceros
\$	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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

### Uso de la Función TO\_CHAR con Números

Al trabajar con valores numéricos como cadenas de caracteres, debe convertir esos números al tipo de datos de carácter mediante la función TO\_CHAR, que traduce un valor del tipo de datos NUMBER al tipo de datos VARCHAR2. Esta técnica es especialmente útil con la concatenación.

## Uso de la Función TO\_CHAR con Números (continuación)

### Elementos de Formato Numérico

Si está convirtiendo un número al tipo de datos de carácter, puede utilizar estos elementos de formato:

Elemento	Descripción	Ejemplo	Resultado
9	Posición numérica (el número de nueves determina el ancho de la visualización)	999999	1234
0	Visualización de ceros iniciales	099999	001234
\$	Signo de dólar flotante	\$999999	\$1234
L	Símbolo de divisa local flotante	L999999	FF1234
D	Devuelve el carácter decimal en la posición especificada. El valor por defecto es un punto (.).	99,99	99D99
.	Punto decimal en la posición especificada	999999,99	1234,00
G	Devuelve el separador de grupos en la posición especificada. Puede especificar varios separadores de grupo en un modelo de formato umérico.	9,999	9G999
,	Coma en la posición especificada	999,999	1,234
MI	Signo menos a la derecha (valores negativos)	999999MI	1234-
PR	Números negativos entre corchetes	999999PR	<1234>
EEEE	Notación científica (el formato debe especificar cuatro letras E)	99,999EEEE	1,234E+03
U	Devuelve la divisa dual "euro" (u otra) en la posición especificada	U9999	€1234
V	Multiplica por 10 <i>n</i> veces ( <i>n</i> = número de nueves tras V)	9999V99	123400
S	Devuelve el valor negativo o positivo	S9999	-1234 o +1234
B	Visualiza los valores cero como espacios en blanco, 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Instrucciones

- Oracle Server muestra una cadena de signos numéricos (#) en lugar de un número completo 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 posiciones decimales que se proporciona en el modelo de formato.



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

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

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

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

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

- **Estas funciones tienen un modificador fx. Este modificador especifica la correspondencia exacta del argumento de carácter y el modelo de formato de fecha de una función TO\_DATE.**

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

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

En ocasiones, deberá convertir una cadena de caracteres a número o a fecha. Para ello, utilice las funciones TO\_NUMBER o TO\_DATE. El modelo de formato que elija se basará en los elementos de formato demostrados anteriormente.

El modificador fx especifica la correspondencia exacta del argumento de carácter y el modelo de formato de fecha de una función TO\_DATE.

- La puntuación y el texto entre comillas del argumento de carácter debe corresponder exactamente (excepto en las mayúsculas/minúsculas) con las partes correspondientes del modelo de formato.
- El argumento de carácter no puede contener espacios en blanco adicionales. Sin fx, Oracle ignora los espacios en blanco adicionales.
- Los datos numéricos del argumento de carácter debe tener el mismo número de dígitos que el elemento correspondiente del modelo de formato. Sin fx, los números del argumento de carácter pueden omitir los ceros iniciales.

## Uso de las Funciones TO\_NUMBER y TO\_DATE (continuación)

### Ejemplo

Muestre el nombre y las fechas de contratación de todos los empleados que comenzaron a trabajar el 24 de mayo de 1999. Como se utiliza el modificador `fx`, se requiere una correspondencia exacta y los espacios tras 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 los dos dígitos especificados del año son:	
		0–49	50–99
Si los dos dígitos del año actual son:	0–49	La fecha devuelta está en el siglo actual	La fecha devuelta está en el siglo anterior al actual
	50–99	La fecha devuelta está en el siglo posterior al actual	La fecha devuelta está en el siglo actual

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

### Elemento de Formato de Fecha RR

El formato de fecha RR es parecido al elemento YY, pero lo puede utilizar para especificar siglos diferentes. Utilice el elemento de formato RR en lugar de YY para que el siglo del valor de retorno varíe según el año de dos dígitos especificado y los dos dígitos del año actual. La tabla de la diapositiva resume el comportamiento del elemento RR.

Año Actual	Fecha Dada	Interpretación (RR)	Interpretación (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 encontrar empleados contratados antes de 1990, utilice el formato RR, que produce los mismos resultados si el comando se ejecuta 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

Copyright © 2004, Oracle. Todos los derechos reservados.

### Ejemplo de Formato de Fecha RR

Para buscar empleados contratados antes de 1990, se puede utilizar el formato RR. Como el año actual es mayor que 1999, el formato RR interpreta la porción del año de la fecha de 1950 a 1999.

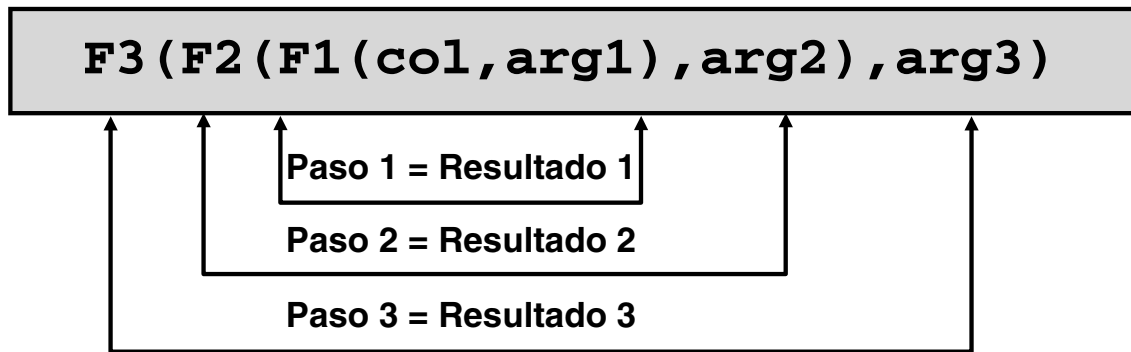
El comando siguiente, por otro lado, da como resultado que no se seleccione ninguna fila porque el formato YY interpreta la porción de la fecha en el siglo actual (2090).

```
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

## Anidamiento de Funciones

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



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

### Anidamiento de Funciones

Las funciones de una sola fila se pueden anidar hasta cualquier profundidad. Las funciones anidadas se evalúan desde el nivel más interno al más externo. A continuación, se ofrecen varios ejemplos de la flexibilidad de estas funciones:

# Anidamiento de Funciones

```
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
Hunold	HUNOLD_US
Ernst	ERNST_US
Lorentz	LORENTZ_US

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Anidamiento de Funciones (continuación)

El ejemplo de la transferencia muestra los apellidos de los empleados del departamento 60. La evaluación de la sentencia SQL implica tres pasos:

1. La función interna recupera los primeros ocho caracteres del apellido.  
Result1 = SUBSTR (LAST\_NAME, 1, 8)
2. La función externa concatena el resultado con \_US.  
Result2 = CONCAT(Result1, '\_US')
3. La función externa convierte los resultados a mayúsculas.

Toda la expresión se convierte en la cabecera de la columna porque no se ha proporcionado alias de columna.

### Ejemplo

Muestra la fecha del siguiente viernes seis meses después de 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 pueden utilizar cualquier tipo de datos y están relacionadas con el uso de valores nulos:

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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Funciones Generales

Estas funciones pueden utilizar cualquier tipo de datos y están relacionados con el uso de valores nulos en la lista de expresiones.

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

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

# Función NVL

**Convierte un valor nulo en un valor real:**

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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Función NVL

Para convertir un valor nulo en un valor 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 datos, pero el valor de retorno es siempre el mismo que el tipo de datos de *expr1*.

### Conversiones NVL para Varios Tipos de Datos

Tipo de Datos	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.

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función NVL

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

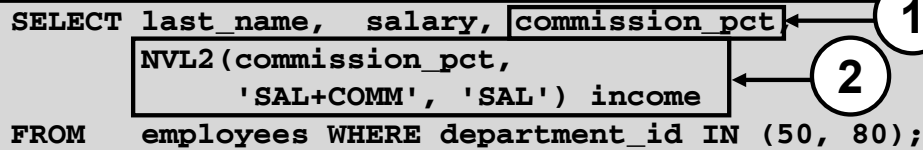
```
SELECT last_name, salary, commission_pct,
      (salary*12) + (salary*12*commission_pct) AN_SAL
FROM employees;
```

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

Observe que la compensación anual se calcula únicamente para los empleados que ganen comisión. Si el valor de alguna columna de una expresión es nulo, el resultado es 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 diapositiva, la función NVL se utiliza para convertir valores nulos en cero.

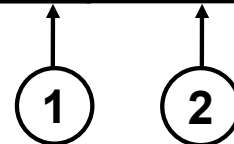
# Uso de la Función NVL2

```
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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función NVL2

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

### Sintaxis

`NVL2(expr1, expr2, expr3)`

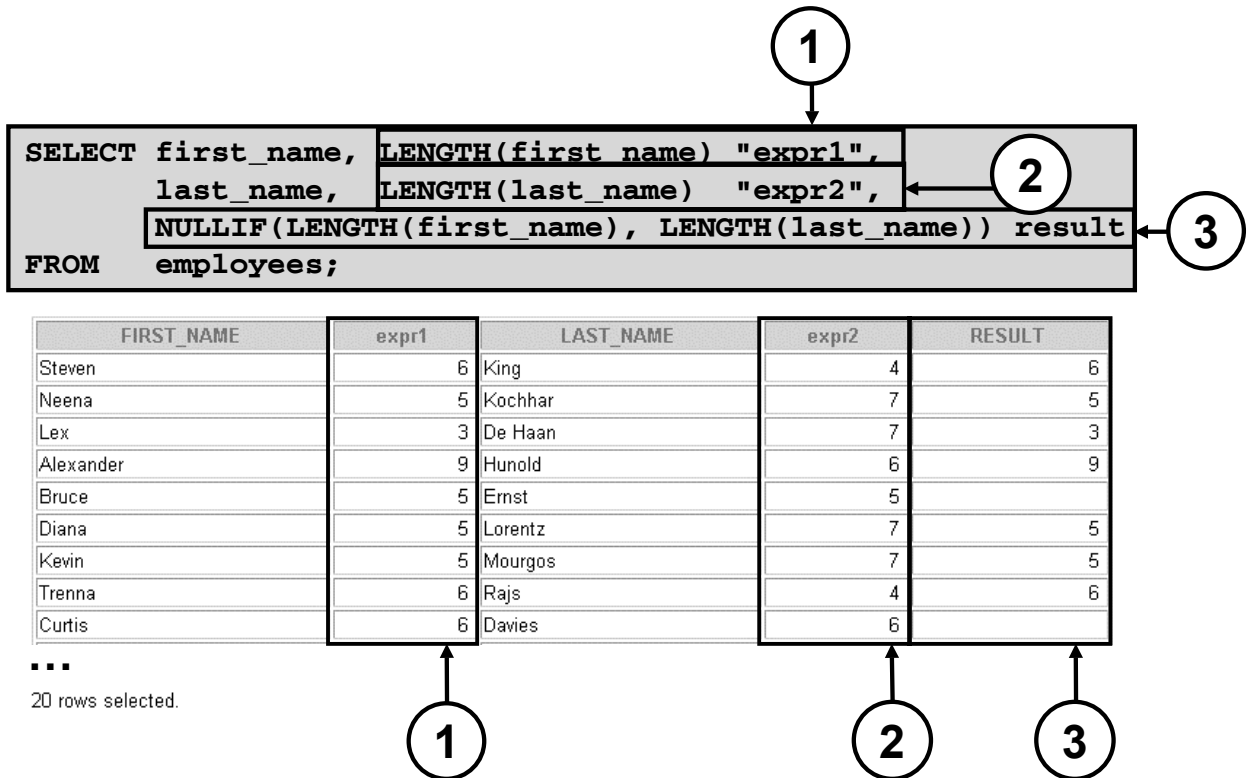
En la sintaxis:

- *expr1* es el valor o la expresión de origen que puede contener un valor nulo
- *expr2* es el valor que se devuelve si *expr1* no es nulo
- *expr3* es el valor que se devuelve si *expr2* es nulo

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

El argumento *expr1* puede tener cualquier tipo de datos. Los argumentos *expr2* y *expr3* pueden tener cualquier tipo de datos excepto LONG. Si los tipos de datos de *expr2* y *expr3* son diferentes, Oracle Server convierte *expr3* al tipo de datos de *expr2* antes de compararlos a menos que *expr3* sea una constante nula. En el último caso, no es necesaria una conversión del tipo de datos. El tipo de datos del valor de retorno es siempre el mismo que el tipo de datos de *expr2*, a menos que *expr2* sean datos de carácter, en cuyo caso el tipo de datos del valor de retorno es VARCHAR2.

# Uso de la Función NULLIF



ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función NULLIF

La función NULLIF compara dos expresiones. Si son iguales, la función devuelve un valor nulo. Si no son iguales, la función devuelve la primera expresión. No 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 que *expr1*, se devuelve *expr1*.)

En el ejemplo que se muestra en la diapositiva, la longitud del nombre de la tabla EMPLOYEES se compara con el apellido de la tabla EMPLOYEES. Si las longitudes del nombre y el apellido son iguales, se devuelve un valor nulo. Si las longitudes del nombre y el apellido no son iguales, se muestra la longitud del nombre.

**Nota:** La función NULLIF es lógicamente equivalente a esta expresión CASE. La expresión CASE se analiza en una página posterior:

```
CASE WHEN expr1 = expr2 THEN NULL ELSE expr1 END
```

# Uso de la Función COALESCE

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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función COALESCE

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

### Sintaxis

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

En la sintaxis:

- *expr1* devuelve esta expresión si no es nula
- *expr2* devuelve esta expresión si la primera expresión es nula y esta expresión no lo es
- *exprn* devuelve esta expresión si las expresiones precedentes son nulas

Todas las expresiones deben ser del mismo tipo de datos.

# Uso de la Función COALESCE

```
SELECT last_name,  
       COALESCE(manager_id,commission_pct, -1) comm  
FROM   employees  
ORDER BY commission_pct;
```

LAST_NAME	COMM
Grant	149
Zlotkey	100
Taylor	149
Abel	149
King	-1
Kochhar	100
De Haan	100

...

20 rows selected.

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función COALESCE (continuación)

En el ejemplo de la diapositiva, si el valor `MANAGER_ID` no es nulo, se muestra. Si el valor `MANAGER_ID` es nulo, se muestra `COMMISSION_PCT`. Si los valores `MANAGER_ID` y `COMMISSION_PCT` son nulos, se muestra el valor `-1`.

# Expresiones Condicionales

- **Permiten utilizar la lógica IF-THEN-ELSE dentro de una sentencia SQL**
- **Puede usar dos métodos:**
  - **Expresión CASE**
  - **Función DECODE**

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Expresiones Condicionales

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

**Nota:** La expresión CASE cumple con ANSI SQL. La función DECODE es específica de la sintaxis Oracle.

# 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Expresión CASE

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

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

Todas las expresiones (*expr*, *comparison\_expr* y *return\_expr*) deben ser del mismo tipo de datos, 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Expresión CASE

En la sentencia SQL de la diapositiva, se descodifica el valor de JOB\_ID. Si JOB\_ID es IT\_PROG, el aumento de salario es del 10 %; si JOB\_ID es ST\_CLERK, el aumento de salario es del 15 %; si JOB\_ID es SA\_REP, el aumento de salario es del 20 %. Para el resto de roles de trabajo, no hay aumento de salario.

Se puede escribir la misma sentencia con la función DECODE.

Esto es un ejemplo de expresión CASE buscada. En una expresión CASE buscada, la búsqueda se produce de izquierda a derecha hasta que se encuentre una incidencia de la condición mostrada y, entonces, devuelve la expresión de retorno. Si no se encuentra ninguna condición que sea verdadera y si existe una cláusula ELSE, se devuelve la expresión de retorno de la cláusula ELSE; de lo contrario, se devuelve NULL.

```
SELECT last_name, salary,
      (CASE WHEN salary<5000 THEN 'Low'
            WHEN salary<10000 THEN 'Medium'
            WHEN salary<20000 THEN 'Good'
            ELSE 'Excellent'
      END) qualified_salary
FROM employees;
```



# Función DECODE

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

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

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Función DECODE

La función DECODE descodifica una expresión de forma parecida a la lógica IF-THEN-ELSE que se utiliza en varios lenguajes. La función DECODE descodifica *expression* tras compararla con cada valor *search*. Si la expresión es igual que *search*, se devuelve *result*.

Si se omite el valor por defecto, se devuelve un valor nulo donde un valor de búsqueda no corresponda a ninguno de los valores del resultado.

# 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

Copyright © 2004, Oracle. Todos los derechos reservados.

## Uso de la Función DECODE

En la sentencia SQL de la diapositiva, se prueba el valor de JOB\_ID. Si JOB\_ID es IT\_PROG, el aumento de salario es del 10 %; si JOB\_ID es ST\_CLERK, el aumento de salario es del 15 %; si JOB\_ID es SA\_REP, el aumento de salario es del 20 %. Para el resto de roles de trabajo, no hay aumento de salario.

Se puede expresar la misma sentencia 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 la tasa de impuestos 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

Copyright © 2004, Oracle. Todos los derechos reservados.

### Uso de la Función DECODE (continuación)

La diapositiva muestra otro ejemplo que utiliza la función DECODE. En este ejemplo, determinamos la tasa de impuestos para cada empleado del departamento 80 basándonos en el salario mensual. Las tasas de impuestos son:

<i>Rango de Salarios Mensuales</i>	<i>Tasa de Impuestos</i>
\$0.00–1,999.99	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 más	45 %

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

# Resumen

**En esta lección ha aprendido a:**

- **Realizar cálculos en datos mediante funciones**
- **Modificar elementos de datos individuales mediante funciones**
- **Manipular la salida para grupos de filas mediante funciones**
- **Modificar formatos de fecha para su visualización mediante funciones**
- **Convertir tipos de datos de columnas mediante funciones**
- **Utilizar funciones NVL**
- **Utilizar la lógica IF-THEN-ELSE**

ORACLE

Copyright © 2004, Oracle. Todos los derechos reservados.

## Resumen

Las funciones de una sola fila se pueden anidar hasta cualquier nivel. Las funciones de una sola fila pueden manipular:

- Datos de carácter: 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

Recuerde que:

- Los valores de datos también pueden utilizar operadores aritméticos.
- Las funciones de conversión pueden convertir valores de carácter, de fecha y numéricos: TO\_CHAR, TO\_DATE, TO\_NUMBER
- Hay varias funciones relacionadas con los valores nulos, como NVL, NVL2, NULLIF y COALESCE.
- Se puede aplicar la lógica IF-THEN-ELSE dentro de una sentencia SQL mediante la expresión CASE o la función DECODE.

## **SYSDATE y DUAL**

SYSDATE es una función de fecha que devuelve la fecha y la hora actuales. Se suele seleccionar SYSDATE en una tabla ficticia denominada DUAL.

## **Práctica 3: Visión General de la Parte 2**

**Esta práctica cubre los temas siguientes:**

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

**ORACLE**

Copyright © 2004, Oracle. Todos los derechos reservados.

### **Práctica 3: Visión General de la Parte 2**

La práctica de la Parte 2 de esta lección proporciona varios ejercicios que utilizan diferentes funciones disponibles para los tipos de datos de carácter, numérico y de fecha. Para la Parte 2, realice los ejercicios 7–14.

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

## Práctica 3

### Parte 1

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

Date
31-DEC-03

2. El departamento de recursos humanos necesita mostrar el número de empleado, el apellido, el salario y el salario aumentado en un 15,5 % (expresado como número entero) de cada empleado. Etiquete la columna como `New Salary`. Guarde la sentencia SQL en un archivo de texto denominado `lab_03_02.sql`.
3. Ejecute la consulta del archivo `lab_03_02.sql`.

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary
100	King	24000	27720
101	Kochhar	17000	19635
...			
202	Fay	6000	6930
205	Higgins	12000	13860
206	Gietz	8300	9587

20 rows selected.

4. Modifique la consulta `lab_03_02.sql` para agregar una columna que reste el antiguo salario al nuevo salario. Etiquete la columna como `Increase`. Guarde el contenido en un archivo denominado `lab_03_04.sql`. Ejecute la consulta revisada.

EMPLOYEE_ID	LAST_NAME	SALARY	New Salary	Increase
100	King	24000	27720	3720
101	Kochhar	17000	19635	2635
102	De Haan	17000	19635	2635
...				
202	Fay	6000	6930	930
205	Higgins	12000	13860	1860
206	Gietz	8300	9587	1287

20 rows selected.

### Práctica 3 (continuación)

5. Escriba una consulta que muestre el apellido (con la primera letra en mayúsculas y las demás en minúsculas) y la longitud del apellido de todos los empleados cuyo nombre comience por *J*, *A* o *M*. Etiquete cada columna de forma adecuada. Ordene los resultados por los apellidos de los empleados.

Name	Length
Abel	4
Matos	5
Mourgos	7

Reescriba la consulta para que se pida al usuario que introduzca la primera letra de un apellido. Por ejemplo, si el usuario introduce *H* cuando se le pida una letra, la salida debería mostrar todos los empleados cuyo apellido comience por la letra *H*.

Name	Length
Hartstein	9
Higgins	7
Hunold	6

### Práctica 3 (continuación)

6. El departamento de recursos humanos desea averiguar el tiempo que llevan contratados todos los empleados. Para cada empleado, muestre el apellido y calcule el número de meses entre hoy y la fecha en que se contrató. Etiquete la columna como MONTHS\_WORKED. Ordene los resultados por el número de meses empleado. Redondee el número de meses al número entero más cercano.

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

LAST_NAME	MONTHS_WORKED
Zlotkey	47
Mourgos	50
Grant	55
Lorentz	59
Vargas	66
Taylor	69
Matos	70
Fay	76
Davies	83
Abel	92
Hartstein	94
Rajs	98
Higgins	115
Gietz	115
De Haan	132
Ernst	151
Hunold	168
Kochhar	171
Whalen	195
King	198

20 rows selected.



### Práctica 3 (continuación)

#### Parte 2

7. Cree un informe que cree 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. ....
...	
Hartstein	earns \$13,000.00 monthly but wants \$39,000.00.
Fay	earns \$6,000.00 monthly but wants \$18,000.00. ....
Higgins	earns \$12,000.00 monthly but wants \$36,000.00. ....
Gietz	earns \$8,300.00 monthly but wants \$24,900.00. ....
20 rows selected.	

Si le queda 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 una longitud de 15 caracteres, rellenos a la izquierda con el símbolo \$. Etiquete la columna como SALARY.

LAST_NAME	SALARY
King	\$\$\$\$\$\$\$\$\$24000
Kochhar	\$\$\$\$\$\$\$\$\$17000
De Haan	\$\$\$\$\$\$\$\$\$17000
Hunold	\$\$\$\$\$\$\$\$\$9000
...	
Fay	\$\$\$\$\$\$\$\$\$6000
Higgins	\$\$\$\$\$\$\$\$\$12000
Gietz	\$\$\$\$\$\$\$\$\$8300
20 rows selected.	

### Práctica 3 (continuación)

9. Muestre el apellido, la fecha de contratación y la fecha de revisión salarial de cada empleado, que es el primer lunes después de seis meses de servicio. Etiquete la columna como 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
...		
Higgins	07-JUN-94	Monday, the Twelfth of December, 1994
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 que empezó el empleado. Etiquete la columna como DAY. Ordene los resultados por día de la semana, empezando 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
...		
Lorentz	07-FEB-99	SUNDAY
Fay	17-AUG-97	SUNDAY
Matos	15-MAR-98	SUNDAY

20 rows selected.

### Práctica 3 (continuación)

Si desea afrontar un desafío mayor, realice estos ejercicios:

11. Cree una consulta que muestre los apellidos y los importes de comisión de los empleados. Si un empleado no gana ninguna comisión, muestre “No Commission”. Etiquete la columna como COMM.

LAST_NAME	COMM
King	No Commission
Kochhar	No Commission
...	...
Zlotkey	.2
Abel	.3
Taylor	.2
Grant	.15
Whalen	No Commission
Hartstein	No Commission
Fay	No Commission
Higgins	No Commission
Gietz	No Commission

20 rows selected.

12. Cree una consulta que muestre los ocho primeros caracteres de los apellidos de los empleados e indique los importes de sus salarios con asteriscos. Cada asterisco representa mil dólares. Clasifique los datos en orden descendente de salario. Etiquete la columna como EMPLOYEES\_AND\_THEIR\_SALARIES.

EMPLOYEES_AND_THEIR_SALARIES
King *****
Kochhar *****
De Haan *****
Hartstei *****
Higgins *****
...
Matos **
Vargas **

20 rows selected.

### Práctica 3 (continuación)

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

<i>Puesto</i>	<i>Grado</i>
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E
Ninguno de las anteriores	0

JOB_ID	GRA
AC_ACCOUNT	0
AC_MGR	0
AD_ASST	0
AD_PRES	A
AD_VP	0
AD_VP	0
IT_PROG	C
IT_PROG	C
IT_PROG	C
MK_MAN	0
MK_REP	0
SA_MAN	0
SA_REP	D
SA_REP	D
SA_REP	D
ST_CLERK	E
ST_CLERK	E
ST_CLERK	E
ST_CLERK	E
ST_MAN	B

20 rows selected.

14. Vuelva a escribir la sentencia del ejercicio anterior mediante la sintaxis CASE.