3.- FUNCIONES A NIVEL DE FILA

3.1. Tipos de funciones SQL.
3.2. Funciones a nivel fila.
3.3. Funciones de caracteres.
- ❖ Funciones filtro.
- ❖ Funciones de manipulación de caracteres.

3.4. Funciones numéricas.
3.5. Funciones de fecha.
3.6. Funciones de conversión.
- ❖ Implícita al tipo de dato.
- ❖ Explicita al tipo de dato.
  - TO_ CHAR
  - TO_ NUMBER Y TO_ DATE

3.7. Funciones generales.
- ❖ NVL
- ❖ DECODE

3.8. Funciones anidadas.

3.1.- TIPOS DE FUNCIONES SQL

Las funciones constituyen una utilidad muy buena dentro de SQL y se pueden utilizar para:

- Realizar cálculos en general sobre datos.
- Modificar ítem de datos individuales.
- Manipular la salida de grupos de fila.
- Modificar formatos de los datos para su presentación.
- Convertir los tipos de datos de las columnas.

Existen dos tipos de funciones en SQL:

1. **Funciones a nivel fila**: Devuelven un valor por cada fila seleccionada en la tabla, pueden ser de carácter, numéricas, de fecha, de conversión y generales.
2. **Funciones a nivel de grupo**: Devuelven un valor por un grupo de filas.

3.2.- FUNCIONES A NIVEL DE FILA

Sintaxis

```
Nb_Función (columna | expresión,[arg1, arg2, ..., arg n])
```

Las funciones a nivel fila:

- ❖ Manipulan ítems de datos.
- ❖ Aceptan argumentos y devuelven un valor.
- ❖ Actúan sobre cada fila retornada.
- ❖ Devuelven un resultado por fila.
- ❖ Modifican el tipo de datos
- ❖ Pueden estar anidadas.

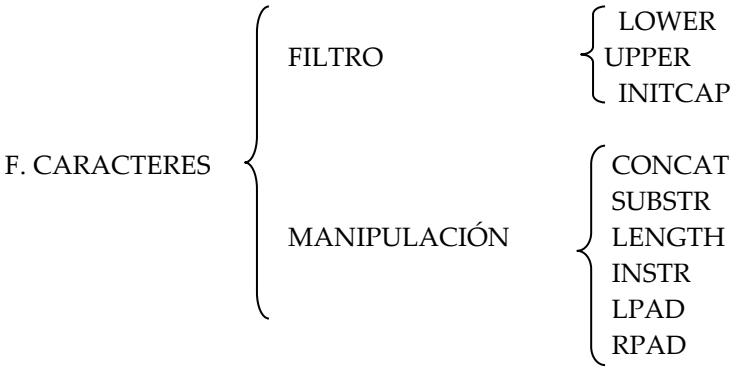
❖ Se pueden usar en las clausulas SELECT, WHERE y 0RDER BY.

Las funciones a nivel fila pueden ser:

- Funciones de caracteres: Aceptan caracteres como datos de entrada y pueden devolver caracteres o números.
- Funciones numéricas: Aceptan números como datos de entrada y devuelven valores numéricos.
- Funciones fecha: Operan con valores de dato tipo fecha.
- Funciones de conversión: Convierten un valor de un tipo de dato.
- Funciones generales.

3.3.- FUNCIONES DE CARACTERES

Aceptan caracteres como datos de entrada y pueden devolver caracteres o números.



❖ FUNCIONES DE FILTRO

LOWER (columna | expresión): Convierte en minúsculas las cadenas de caracteres mayúsculas o mixtas.

UPPER (columna | expresión): Convierte a mayúsculas las cadenas de caracteres minúsculas o mixtas

INITCAP (columna | expresión): Convierte la primera letra de una cadena en mayúsculas u el resto en minúsculas

Ejemplo de conversión de cadenas de caracteres:

Función	**Resultado**
LOWER('Curso SQL')	curso sql
UPPER('Curso SQL')	CURSO SQL
INITCAP('Curso SQL')	Curso Sql

Ejemplo 1:

```
SELECT 'El trabajo de ' || INITCAP(ENAME)||'es'|| LOWER(JOB) "Oficio"
FROM EMP;
```

Ejemplo 2:

```
SELECT EMPNO, ENAME, DEPTNO
FROM EMP
WHERE ENAME = UPPER('blake');
```

❖ FUNCIONES DE MANIPULACIÓN DE CARACTERES.

CONCAT(columna1|expresión1, columna2|expresión2, ...): Concatena la primera columna de valores con la segunda, siendo equivalente a ||.

SUBSTR(columna | expresión, m [,n]) : Extrae una cadena de una determinada medida según se especifiquen sus argumentos.

LENGTH(columna | expresión): muestra la longitud de una cadena como un valor numérico

INSTR(columna | expresión, m): Encuentra la posición numérica de un carácter nombrado

LPAD(columna | expresión, n, 'string'): Justifica a la izquierda el valor del carácter.

RPAD(columna | expresión, n, 'string'): Justifica a la derecha el valor del carácter.

Ejemplo de conversión de cadenas de caracteres:

Función	**Resultado**
CONCAT('hola','cepillo')	holacepillo
SUBSTR('ARMARIO',1,4)	ARMA
LENGTH('ARMARIO')	7
INSTR ('ARMARIO', 'M')	3
LPAD (SAL,8, '*')	****5000
RPAD (SAL,8, '*')	5000****

Ejemplo 1:

Visualiza el nombre del empleado junto con el oficio, la longitud del nombre del empleado y la posición numérica de la letra 'A' del nombre del empleado, para todos los empleados que están en el departamento 30

```
SELECT      ENAME, CONCAT(ENAME, JOB ), LENGTH(ENAME),
            INSTR(ENAME, 'A')
FROM EMP
WHERE DEPTNO = 30;
```

Ejemplo 2:

Visualiza todos los empleados en los que su nombre acabe en N

```
SELECT ENAME
FROM EMP
WHERE  SUBSTR(ENAME,-1,1)= 'N';
```

Ejemplo 3:

Representar el salario de un empleado con * por cada 100$ que cobra.

```
SELECT RPAD(ENAME,8)|| RPAD(' ',SAL/100,'*') HISTOGRAMA
FROM EMP;

HISTOGRAMA
----------------------------------------------------------------
SMITH   *******
ALLEN   **************
WARD    **********
JONES   *************************
MARTIN  **********
BLAKE   *************************
CLARK   ***********************
SCOTT   ****************************
KING    ***********************************************
TURNER  *************
ADAMS   *********
JAMES   ********
FORD    ****************************
MILLER  ***********
```

3.4.- FUNCIONES NUMÉRICAS

ROUND(columna | expresión, n): Redondea la columna o expresión a un valor n posiciones decimales. Si se omite n no se redondea con lugares decimales, y si n es negativo los números a la izquierda del punto decimal se redondean.

Ejemplo 1:

Visualizar el valor 45.923 redondeado a centenas, 0 y 10 posiciones decimales.

```
SELECT ROUND(45.932,2), ROUND(45.932,0), ROUND(45.932,-1)
FROM   SYS.DUAL;
```

Los valores que devuelve son por tanto los siguientes:

ROUND(45.923,2)→45.92
ROUND(45.923) → 46
ROUND(45.923,-1)→ 50

TRUNC(columna | expresión, n): Elimina los dígitos indicados. Trunca la columna o valor en la enésima posición decimal, si se omite n, sin lugares decimales si n es negativo los números a la izquierda del punto decimal se truncan a 0

Ejemplo 2:

Visualizar el valor 45.923 con TRUNC a centenas, 0 y 10 posiciones decimales.

```
SELECT      TRUNC(45.932,2), TRUNC(45.936,2), TRUNC(45.932,0),
            TRUNC(45.932,-1)
FROM  SYS.DUAL;
```

Los valores que devuelve son por tanto los siguientes:

TRUNC(45.923,2)→ 45.92
TRUNC(45.936,2) → 45.93
TRUNC(45.923) → 45
TRUNC(45.923,-1)→ 40

MOD(m,n): devuelve el resto de la división entre dos números (m entre n).

Ejemplo 3:

Calcular el resto del salario entre la comisión para todos los empleados cuyo oficio es SALESMAN.

```
SELECT MOD(SAL, COMM)
FROM EMP
WHERE JOB ='SALESMAN';
```

SYS.DUAL es una tabla virtual de la base de datos que mantiene la información necesaria sobre el sistema. Es propiedad del usuario SYS y pueden acceder a ella todos los usuarios. Contiene una columna llamada DUMMY y una fila X. Es útil cuando se quiere devolver un valor una sola vez, por ejemplo un valor constante o expresión que no se deriva a partir de los datos del usuario. Es también muy útil para visualizar fecha y la hora del sistema a través de la función SYSDATE.

3.5.- FUNCIONES DE FECHAS.

Para este tipo de funciones hay que tener en cuenta cual es el formato de la fecha en el sistema. El formato por defecto de la fecha en ORACLE es DD-MON-YY.

SYSDATE es una función que devuelve la fecha y la hora.

OPERADORES ARITMÉTICOS CON FECHAS.

Permiten sumar o restar un número a una fecha dando como resultado otra fecha, en cambio si se restan dos fechas da como resultado la cantidad de días comprendidos entre las dos fechas.

Operación	Resultado	Descripción
Fecha + número	Fecha	Agrega una cantidad de días a la fecha
Fecha – número	Fecha	Resta una cantidad de días a partir de una fecha
Fecha – fecha	Número de días	Resta una fecha de otra
Fecha + número/24	Fecha	Agrega una cantidad de horas a la fecha

Ejemplo:

```
SELECT ENAME, SYSDATE – HIREDATE FROM EMP
WHERE DEPTNO = 10
```

Devuelve los empleados y el número de días que hay entre la fecha del sistema y la fecha de entrada en la empresa. La parte no entera el número que devuelve indica horas.

FUNCIONES DE FECHA

Las funciones de fecha operan sobre fechas ORACLE. Todas ellas devuelven como resultado un valor fecha, salvo MONTHS_BETWEEN que devuelve un valor numérico.

MONTHS _BETWEEN(fecha1,fecha2): Devuelve la cantidad de meses que hay entre la fecha 1 y la fecha 2.

Ejemplo: MONTHS_ BETWEEN(HIREDATE, SYSDATE)

ADD_ MONTHS(fecha, n): agrega número de meses a la fecha.

Ejemplo: ADD_ MONTHS(HIREDATE, 3)

NEXT_DAY(fecha, 'carácter')**:** devuelve la fecha del día, especificado en carácter, siguiente a la fecha, carácter puede ser un número o un literal

Ejemplo: NEXT _DAY (HIREDATE, 'LUNES')
NEXT _DAY(HIREDATE, 1)

LAST_ DAY(fecha): devuelve la fecha del último día del mes que contiene la fecha.

Ejemplo: LAST_DAY(HIREDATE)

ROUND(fecha [,'fmt']): Cuando no se especifica ningún formato devuelve la fecha del primer día del mes contenido en la fecha. Si el formato es YEAR, redondea al año.

TRUNC(fecha [,'fmt']):Escribe el primer DIA del mes, (si no se especifica formato, o del año (si el formato es YEAR), que contenga la fecha-

Función	**Resultado**
MONTHS_BETWEEN('01-SEP-95','11-JAN-94')	19.6774194
ADD_ MONTHS('11-ENE-94',6)	'11-JUL-94'
NEXT_ DAY('01-SEP-95', 'VIERNES')	'02-SEP-95'
LAST_DAY('01-SEP-95')	'30-SEP-95'
ROUND('25-JUL-95','MONTH')	'01-AGO-95'
ROUND('25-JUL-95','YEAR')	'01-ENE-96'
TRUNC('25-JUL-95','MONTH')	'01-JUL-95'
TRUNC('25-JUL-95','YEAR')	'01-ENE-95'

3.6.- FUNCIONES DE CONVERSIÓN.

Existen dos tipos de funciones de conversión: implícitas y explicitas. Las implícitas son las realizadas automáticamente por ORACLE

❖ IMPLÍCITAS AL TIPO DE DATO:

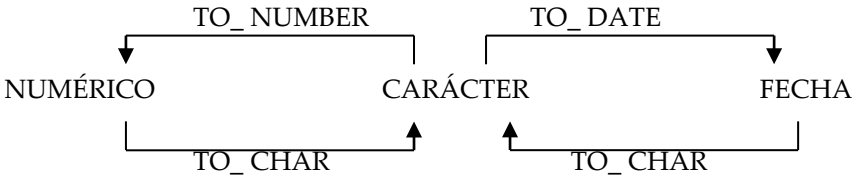
ORACLE puede utilizar tipos de datos de ANSI, DB2 y SQL/DS, convirtiendo estos tipos de datos en datos propios. Las asignaciones que Oracle 8 convierte automáticamente se les llaman implícitas al tipo de dato, así pues puede convertir automáticamente:

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

❖ **EXPLICITAS AL TIPO DE DATO**.

Son los datos que pueden ser modificaros o cambiados de formato por el usuario que maneja la base de datos. ORACLE tiene una serie de funciones de conversión específicas:

- TO_NUMBER
- TO_DATE
- TO_CHAR



- **TO_ CHAR**

➢ **Fecha a cadena**

Convierte una **fecha** en una cadena de **caracteres** VARCHAR2.

Sintaxis:

```
TO_ CHAR (fecha [, 'fmt'])
```

FMT es sensible a las mayúsculas y a las minúsculas, puede incluir cualquier elemento de formato válido, es decir, cualquier elemento con formato de fecha.

Tiene un elemento FILL MODE para eliminar espacios en blanco o suprimir cero a la izquierda. El formato debe ir separado de la fecha por una coma y el ancho de esta columna, por defecto es de 80 caracteres.

ELEMENTOS DEL MODELO FORMATO DE FECHA

Y: última cifra del año.
YY: dos últimas cifras del año.
YYY: tres últimas cifras del año.
YYYY: año completo.
YEAR: año en letra.
MM: número de mes con dos dígitos.
MON: Nombre del mes con tres caracteres.
MONTH: Nombre completo del mes.

DY: abreviatura de tres letras del día de la semana.
DAY: nombre del día de la semana completo.
SYEAR: indica si el año es AC o DC.
Q: trimestre del año
RM: mes en números romanos.
WW : semana del año .
W: semana del mes.
DDD: día del año.
DD: día del mes.
D: día de la semana.
J: día juliano (número de días desde el 31-12- 4712 AC).

Ejemplo:

Obtener la hora del sistema:

```
SELECT TO_CHAR(SYSDATE,'HH24:MI:SS')
FROM SYS.DUAL;
```

FORMATOS DE TIEMPO

AM o **PM**: mañana o tarde
A.M. o P.M.: indicador de meridiano con periodos.
HH , **HH12**, **HH24** : indica las horas en 12 o 24
MI: minutos
SS: segundos
SSSS: cuenta los segundos desde media noche.
" ": inclusión de literales.
DDSPTH: sufijo que permite escribir en número ordinal de días en letra.
DDSP: número de día ordinal en letra.

Ejemplo 1:

Visualizar el nombre y la fecha de contratación de los empleados de la empresa. Con el siguiente formato de fecha: 17 DE DICIEMBRE DE 1980.

```
SELECT ENAME, TO_CHAR(HIREDATE, ' DD "DE" MONTH "DE" YYYY')
FROM EMP;
```

Ejemplo 2:

Modificar el ejemplo anterior para visualizar las fechas en el siguiente formato: SEVENTEENTH DE DICIEMBRE DE 1980 12: 00: 00 AM.

```
SELECT ENAME, TO_CHAR(HIREDATE,
                'DDSPTH "DE" MONTH "DE" YYYY " " HH:MI:SS " " AM')
FROM EMP;
```

- **Número a cadena**

Convierte un **número** en una cadena de **caracteres** VARCHAR2.

Sintaxis:

```
TO_ CHAR (número [ , 'fmt'])
```

ELEMENTOS QUE INTERVIENEN EN EL FORMATO

9: representa un número.
0: muestra 0 a la izquierda.
$: signo de dólar de moneda delante del número.
L: símbolo de moneda local.
. : imprime punto decimal.
, : imprime indicador de millar.
MI: signo negativo a la derecha del número.
PR: pone entre paréntesis los números negativos.
E: notación científica.
V: multiplica por 10 el valor, si a su derecha hay un 9, por 100 si hay 99...
B: muestra los ceros como espacios en blanco.

Ejemplo:

Poner los sueldos de los empleados en pesetas, sabiendo que 1$ = 201 Ptas.

```
SELECT ENAME, TO_CHAR(SAL* 201,'0999999L')
FROM EMP;
```

- **TO_NUMBER Y TO_ DATE**

Convierte una cadena de caracteres a un formato numérico o a una fecha respectivamente.

Sintaxis:

```
TO_ NUMBER (char)
TO _ DATE (char [, 'fmt'])
```

El formato de la función TO_ DATE es igual que los formatos de la función TO_CHAR

Ejemplo:

Mostrar los nombres y las fechas de contratación de todos los empleados que se incorporaron el 22 de febrero de 1981.

```
SELECT ENAME, HIREDATE
FROM EMP
WHERE HIREDATE = TO_DATE('FEBRERO 22, 1981', 'Month dd, YYYY');
```

3.7.- FUNCIONES GENERALES

❖ NVL

Se utiliza para convertir los valores nulos a un valor especificado.

Sintaxis:

```
NVL (expresión1, expresión2)
```

Donde:
Expresión1: es el valor de la fuente o expresión que podría contener un nulo.
Expresión2: es el valor en que se va a convertir el nulo.

Los tipos de dato en la función NVL deben coincidir:

```
NVL(HIREDATE,'17-AGO-1999')
NVL(ENAME,'DESCONOCIDO')
NVL(COMM,0)
```

Ejemplo:

Calcular la compensación anual de todos los empleados de la empresa.

```
SELECT ENAME, SAL*1000, COMM,(SAL*12)+NVL(COMM,0) "SUELDO ANUAL"
FROM EMP;
```

❖ DECODE

Es similar a las estructuras CASE o IF_ THEN_ ELSE, para facilitar las consultas condicionales. Esta función descifra una expresión después de compararla con cada valor de búsqueda, si la expresión es la misma que la búsqueda, se devuelve el resultado, en cambio si se omite el valor por defecto, se devolverá un valor nulo donde una búsqueda no coincida con ninguno de los valores resultantes.

Sintaxis:

```
DECODE (columna/expresión,   selección1, resultado1,
                             [selección2, resultado2,...]
                             [, default] )
```

Ejemplo:

Incrementar el salario de los ANALYST un 10%, los CLERK un 15% y los MANAGER un 20%.

```
SELECT JOB, SAL, DECODE(JOB,      'ANALYST',    SAL* 1.1,
                                  'CLERK',      SAL* 1.15,
                                  'MANAGER',    SAL* 1.2,
                                                SAL) "Revisión Salarial"
FROM EMP;
```

3.8.- FUNCIONES ANIDADAS

Las funciones a nivel fila pueden anidarse hasta cualquier nivel y son evaluadas desde el nivel más profundo al nivel menos profundo.

Sintaxis:

```
... F3 (F2 (F1 (column, arg1), arg2 ), arg3) ...
```

Ejemplo:

Visualizar la fecha del próximo viernes que se lleva 6 meses con la fecha de contratación. La fecha resultante debería tener el formato "*viernes , noviembre , twenty-seventh, 1993"*, ordenando los resultados por fecha de contratación, también se creará un alias en el que figure "REVISIÓN PRÓXIMOS 6 MESES"

```
SELECT      ENAME, HIREDATE,
            TO_CHAR (NEXT_DAY (ADD_MONTHS (HIREDATE, 6), 'VIERNES'), 'day,
            month, ddspth, yyyy') "REVISIÓN PRÓXIMOS 6 MESES"
FROM  EMP
ORDER BY 2;
```