

# **Bases de Dados**

Engenharia Informática regimes Diurno e Pós-Laboral 2º Ano 1.º Semestre

# SQL - Folha de apoio

<b>FUNÇÕES I</b>	DA LINGUAGEM SQL EM ORACLE	3
1 FUNÇ	ÕES DE LINHA	3
1.1 F	Funções Numéricas	3
1.1.1	ABS.	
1.1.2	ACOS.	
1.1.3	ATAN	
1.1.4	CEIL	
1.1.5	COS	
1.1.6	COSH	
1.1.7	EXP	
1.1.8	FLOOR	
1.1.9	LN	
1.1.10	LOG	
1.1.11	MOD	
1.1.12	POWER	
1.1.13	ROUND (número)	
1.1.14	SIGN	
1.1.15	SIN	
1.1.16	SINH	
1.1.17	SQRT	
1.1.18	<i>TAN</i>	
1.1.19	TANH	
1.1.20	TRUNC (número)	
	UNÇÕES DE CARACTER	
1.2.1	Funções de Caracter que devolvem um caracter	
1.2.2	Funções de Caracter que devolvem um número	
	PUNÇÕES DE DATA	
1.3.1	ADD MONTHS.	
1.3.2	LAST_DAY	
1.3.3	MONTHS BETWEEN	
1.3.4	NEW TIME	
1.3.5	NEXT_DAY	
1.3.6	NLS_DATE_FORMAT	
1.3.7	ROUND (data)	
1.3.8	SYSDATE	
1.3.9	TRUNC (data)	
1.4 F	UNÇÕES DE CONVERSÃO	
1.4.1	ASCIISTR	18
1.4.2	<i>BIN_TO_NUM</i>	18
1.4.3	CAST	18
1.4.4	COMPOSE	18
1.4.5	CONVERT	19
1.4.6	DECOMPOSE	19
1.4.7	<i>TO_CHAR</i> (caracter)	19
1.4.8	TO_CHAR (data)	20
1.4.9	TO_CHAR (número)	20
1.4.10	TO_CLOB	21
1.4.11	TO_DATE	22
1.4.12	TO_LOB	22
1.4.13	TO_NUMBER	22
1.4.14	UNISTR	23
	Funções Miscelânea	24
1.5.1	BFILENAME	
1.5.2	COALESCE	24

	1.5.3	Expressão CASE	25
	1.5.4	DECODE	
	1.5.5	EMPTY_[B   C]LOB	26
	1.5.6	GREATEST	26
	1.5.7	LEAST	26
	1.5.8	NULLIF	27
	1.5.9	NVL	27
	1.5.10	) NVL2	27
	1.5.11	UID	28
	1.5.12	<i>USER</i>	28
	1.5.13	B USERENV	28
	1.5.14	! VSIZE	28
2	FUNC	ÇÕES DE GRUPO	29
2		•	
2	2.1	Ç <b>ÕES DE GRUPO</b>	29
2	2.1 2.2	AVG	29
2	2.1 2.2 2.3	AVGCOUNT	29 29
2	2.1 2.2 2.3 2.4	AVGCOUNTMAX	29 29 30
2	2.1 2.2 2.3 2.4 2.5	AVG	29 30 30
2	2.1 2.2 2.3 2.4 2.5 2.6	AVG	29 30 30 30
2	2.1 2.2 2.3 2.4 2.5 2.6 2.7	AVG	29 30 30 30 30
2	2.1 2.2 2.3 2.4 2.5 2.6 2.7	AVG	29 30 30 31 31
2	2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8	AVG	29 30 30 31 31

# Funções da Linguagem SQL em Oracle

Serve o presente documento para descrever a sintaxe de algumas funções específicas da linguagem SQL tal como existe no *Oracle 9i*.

As funções devem ser invocadas em comandos SQL ou em blocos de código PL/SQL.

# 1 Funções de linha

As funções de linha têm as seguintes características:

- Actuam sobre cada registo
- Produzem apenas um valor por registo
- Podem receber um ou mais argumento de entrada
- Podem ser encadeadas
- Podem ser utilizadas onde se utilizam colunas, expressões ou cláusulas SELECT, WHERE e ORDER BY.

Quando se pretende chamar uma função de linha que não está relacionada com nenhuma tabela, em particular, deve-se usar a tabela **DUAL**. Por exemplo:

```
SELECT sysdate FROM DUAL;
```

# 1.1 Funções Numéricas

# 1.1.1 ABS

ABS (col | num)

Devolve o valor absoluto da coluna ou expressão numérica.

#### Exemplo:

Apresente o valor absoluto de -15.

```
SELECT ABS(-15) "Absoluto" FROM DUAL;
```



### 1.1.2 ACOS

ACOS (col | num)

Devolve o arco coseno de um número. O argumento tem de estar no intervalo [-1,1], e a função retorna um valor no intervalo  $[0, \Pi]$ , expresso em radianos.

#### Exemplo:

Apresente o arco coseno de 0.3.

EIBD: funções Oracle Página 3 de 31

SELECT ACOS(0.3) "Arco coseno" FROM DUAL;

```
Arco coseno
------
1.26610367
```

# 1.1.3 ATAN

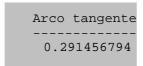
ATAN (col | num)

Devolve o arco tangente de um número. A função retorna um valor no intervalo [- $\Pi$ /2,  $\Pi$ /2], expresso em radianos.

# Exemplo:

Apresente o arco tangente de 0.3.

SELECT ATAN(0.3) "Arco tangente" FROM DUAL;



# 1.1.4 **CEIL**

CEIL (col | num)

Devolve o menor inteiro que seja maior ou igual ao parâmetro de entrada. Ver FLOOR.

# Exemplo:

Apresente o menor inteiro, maior ou igual a 15.7.

SELECT CEIL(15.7) "CEIL" FROM DUAL;



# 1.1.5 COS

COS (col | num)

Devolve o coseno de um número (ângulo expresso em radianos).

# Exemplo:

Apresente o coseno de 180 graus.

SELECT COS(180 \* 3.14159265359 / 180) "Coseno"
FROM DUAL;



# 1.1.6 COSH

COSH (col | num)

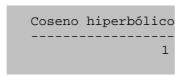
EIBD: funções Oracle Página 4 de 31

Devolve o coseno hiperbólico de um número.

# Exemplo:

Apresente o coseno hiperbólico de 0.

SELECT COSH(0) "Coseno hiperbólico" FROM DUAL;



# 1.1.7 EXP

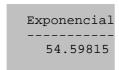
EXP (col | num)

Devolve a exponencial de um número.

# Exemplo:

Apresente o exponencial de 4.

SELECT EXP(4) "Exponencial" FROM DUAL;



#### 1.1.8 FLOOR

FLOOR (col | num)

Devolve o maior inteiro que seja menor ou igual ao parâmetro de entrada. Ver CEIL.

#### Exemplo:

Apresente o maior inteiro que seja menor ou igual a 15.7.

SELECT FLOOR(15.7) "FLOOR" FROM DUAL;



# 1.1.9 LN

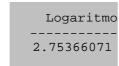
LN (col | num)

Devolve o logaritmo de um número. O número recebido tem de ser superior a 0.

# Exemplo:

Apresente o logaritmo de 95.

SELECT LN(15.7) "Logaritmo" FROM DUAL;



### 1.1.10 LOG

LOG (col1 | num1, col2 | num2)

EIBD: funções Oracle Página 5 de 31

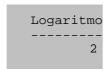


Devolve o logaritmo base m de n. A base m tem de ser um número positivo diferente de m0 e de m1. O m2 pode ser qualquer número positivo.

# Exemplo:

Apresente o logaritmo base 10 de 100.

SELECT LOG(10, 100) "Logaritmo" FROM DUAL;



# 1.1.11 MOD

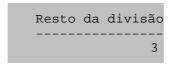
```
LOG (col1 | num1, col2 | num2)
```

Determina o resto da divisão do primeiro parâmetro pelo segundo.

# Exemplo:

Apresente o resto da divisão de 11 por 4.

SELECT MOD(11,4) "Resto da divisão" FROM DUAL;



## 1.1.12 **POWER**

```
POWER (col | num, n)
```

Eleva a coluna ou expressão à potência de n. n tem que ser inteiro.

### Exemplo:

Apresente a potência de 3 elevado a 2.

SELECT POWER(3, 2) "Potência" FROM DUAL;



# 1.1.13 ROUND (número)

```
ROUND (col | num, n)
```

Se n for omitido há arredondamento para o inteiro mais próximo.

Se  ${\bf n}$  for positivo há arredondamento até às  ${\bf n}$  casas decimais.

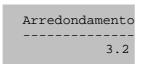
Se  ${\tt n}$  for negativo o arredondamento tem inicio  ${\tt n}$  casas à esquerda da vírgula. Ver TRUNC.

EIBD: funções Oracle Página 6 de 31

# Exemplo:

Apresente o valor arredondado de 3.156 à primeira casa decimal.

SELECT ROUND(3.156, 1) "Arredondamento" FROM DUAL;



# 1.1.14 SIGN

SIGN (col | num)

Devolve -1 se a coluna ou expressão tiverem um valor negativo.

Devolve 1 se tiverem um valor positivo.

Devolve 0 se tiverem um valor de 0.

# Exemplo:

Apresente o valor do sinal de -15.

SELECT SIGN(-15) "Sinal" FROM DUAL;



# 1.1.15 SIN

SIN (col | num)

Devolve o seno de um número (ângulo expresso em radianos).

# Exemplo:

Apresente o seno de 30 graus.

SELECT SIN(30 \* 3.14159265359 / 180) "Seno" FROM DUAL;



### 1.1.16 SINH

SINH (col | num)

Devolve o seno hiperbólico de um número.

# Exemplo:

Apresente o seno hiperbólico de 1.

SELECT SINH(1) "Seno hiperbólico" FROM DUAL;

Seno hiperbólico -----1.17520119

EIBD: funções Oracle Página 7 de 31

# 1.1.17 SQRT

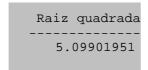
```
SQRT (col | num)
```

Devolve a raiz quadrada da coluna ou expressão.

## Exemplo:

Apresente a raiz quadrada de 26.

```
SELECT SQRT(26) "Raiz quadrada" FROM DUAL;
```



# 1.1.18 TAN

```
TAN (col | num)
```

Devolve a tangente de um número (ângulo expresso em radianos).

## Exemplo:

Apresente a tangente de 135 graus.

```
SELECT TAN(135 * 3.14159265359 / 180) "Tangente"
FROM DUAL;
```



# 1.1.19 TANH

```
TANH (col | num)
```

Devolve a tangente hiperbólica de um número.

#### Exemplo:

Apresente a tangente hiperbólica de 0.5.

```
SELECT TANH(0.5) "Tangente hiperbólica" FROM DUAL;
```

```
Tangente hiperbólico
------
0.462117157
```

# 1.1.20 TRUNC (número)

```
TRUNC (col | num, n)
```

Faz o mesmo que o ROUND mas em vez de arredondar trunca. Ver ROUND.

#### Exemplo:

Apresente o valor truncado de 15.77 à primeira casa decimal.

Truncagem -----15.7

EIBD: funções Oracle Página 8 de 31

# 1.2 Funções de caracter

# 1.2.1 Funções de Caracter que devolvem um caracter

### 1.2.1.1 CHR

CHR (n)

Devolve o caracter com esse número. Ver ASCII.

# Exemplo:

Apresente os caracteres associados aos números 67, 65 e 79.

```
SELECT CHR(67) | CHR(65) | CHR(79) "Animal" FROM DUAL;
```



### 1.2.1.2 **CONCAT**

CONCAT (frase1, frase2)

Junta duas strings ou frases.

# Exemplo:

Apresente a junção de 'BD' com '2'.

```
SELECT CONCAT('BD', '2') "Juntar" FROM DUAL;
```



# 1.2.1.3 INITCAP

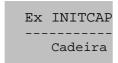
INITCAP (col | string)

Converte a primeira letra para maiúscula e as restantes para minúsculas. Ver LOWER e UPPER.

# Exemplo:

Apresente a palavra 'CADEIRA' com a primeira letra em maiúscula e as restantes em minúsculas.

```
SELECT INITCAP('CADEIRA') "Ex INITCAP" FROM DUAL;
```



### 1.2.1.4 LOWER

```
LOWER (col | string)
```

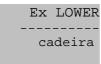
Converte as letras para minúsculas. Ver UPPER e INITCAP.

EIBD: funções Oracle Página 9 de 31

# Exemplo:

Apresente a palavra 'CADEIRA' em minúsculas.

```
SELECT LOWER('CADEIRA') "Ex LOWER" FROM DUAL;
```



### 1.2.1.5 LPAD

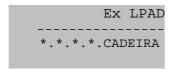
```
LPAD (col | string, n, ['cadeia'])
```

Coloca espaços ou repetições de 'cadeia' à esquerda da *string* até atingir um comprimento de n. Se n for menor que o tamanho da *string* inicial, esta é cortada. Ver RPAD.

## Exemplo:

Apresente o texto '\*.' ao lado esquerdo da palavra 'CADEIRA' até atingir 15 caracteres.

```
SELECT LPAD('CADEIRA', 15, '*.') "Ex LPAD"
FROM DUAL;
```



#### 1.2.1.6 LTRIM

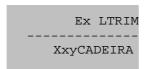
```
LTRIM (col | string, ['car'])
```

Remove todos os espaços ou ocorrências do caracter 'car', à esquerda da *string* de entrada. Ver RTRIM, LPAD e RPAD.

#### Exemplo:

Remova os caracteres 'x' e 'y' do lado esquerdo da palavra 'xyxXxyCADEIRA'.

```
SELECT LTRIM('xyxXxyCADEIRA', 'xy') "Ex LTRIM"
FROM DUAL;
```



# 1.2.1.7 **REPLACE**

```
REPLACE (col | string, cadeia_inical, cadeia_final)
```

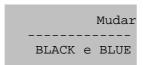
Procura na *string*, *sub-strings* iguais a **cadeia\_inicial** e substitui-as por **cadeia\_final**. Se **cadeia\_final** não for especificada, as *sub-strings* são apenas retiradas.

#### Exemplo:

Substitua as ocorrências de 'J' por 'BL'.

EIBD: funções Oracle Página 10 de 31

SELECT REPLACE('Jack e Jue', 'J', 'BL') "Mudar"
FROM DUAL;



#### 1.2.1.8 RPAD

```
RPAD (col | string, n, ['cadeia'])
```

Devolve uma nova *string* que representa a *string* de entrada concatenada com uma *string* constituída por n repetições da string 'cadeia'. Se 'cadeia' for omitida, então coloca espaços em branco.

Ver LPAD, RTRIM e LTRIM.

# Exemplo:

Coloque o texto '\*.' ao lado direito da palavra 'CADEIRA' até atingir 15 caracteres.

```
SELECT RPAD('CADEIRA', 15, '*.') "CADEIRA" FROM DUAL;
```



#### 1.2.1.9 RTRIM

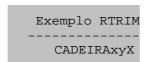
```
RTRIM (col | string, ['car'])
```

Retira todas as ocorrências do caracter 'car', que estejam à direita da *string*. Se 'car' não for especificado, então, retira todos os espaços em branco. Ver LTRIM, LPAD e RPAD.

#### Exemplo:

Remova os caracteres 'x' e 'y' do lado direito da palavra 'CADEIRAxyXxy'.

```
SELECT RTRIM('CADEIRAxyXxy', 'xy') "Exemplo RTRIM"
FROM DUAL;
```



#### 1.2.1.10 SUBSTR

```
SUBSTR (col | valor, pos, [n])
```

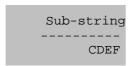
Devolve uma *sub-string* da *string* de entrada. A *substring* começa no caracter **pos** e tem comprimento (facultativo) **n**. Se não for especificado, o comprimento, a *sub-string* começa no caracter **pos** e vai até ao fim da *string* inicial. Ver INSTR.

### Exemplo:

Apresente uma sub-string da frase 'ABCDEFG'.

EIBD: funções Oracle Página 11 de 31

SELECT SUBSTR('ABCDEFG', 3, 4) "Sub-string" FROM DUAL;



#### 1.2.1.11 TRANSLATE

```
TRANSLATE (col | string, de, para)
```

Transforma as ocorrências de caracteres do conjunto **de** nos respectivos caracteres do conjunto **para**. Se **para** não for especificado, retira apenas as ocorrências de **de**.

# Exemplo:



### 1.2.1.12 TRIM

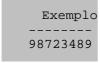
```
TRIM ('car' FROM col | string)
```

Retira todas as ocorrências do caracter 'car' na string de entrada. Se 'car' não for especificado, retira os espaços em branco.

# Exemplo:

Retire os 0s da sequência de números 0009872348900.

```
SELECT TRIM(0 FROM 0009872348900) "Exemplo" FROM DUAL;
```



# 1.2.1.13 UPPER

```
UPPER (col | string)
```

Converte as letras da *string* para maiúsculas. Ver LOWER e INITCAP.

### Exemplo:

Apresente a palavra cadeira em maiúsculas.

SELECT UPPER('cadeira') "Ex UPPER" FROM DUAL;

Ex UPPER -----CADEIRA

EIBD: funções Oracle Página 12 de 31

# 1.2.2 Funções de Caracter que devolvem um número

### 1.2.2.1 ASII

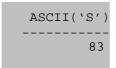
ASCII (char)

Devolve o número que corresponde a esse caracter no código ASCII. Ver CHR.

# Exemplo:

Apresente o número em ASCII do caracter 'S'.

SELECT ASCII('S') FROM DUAL;



#### 1.2.2.2 INSTR

INSTR (col | valor, 'cadeia')

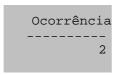
Devolve a posição da primeira ocorrência de 'cadeia' em col ou valor. Ver SUBSTR.

Localiza a posição da n-esima ocorrência de 'cadeia' a partir da posição pos em col ou valor.

# Exemplo:

Apresente a primeira ocorrência da letra 'a' na palavra 'Cadeira'.

SELECT INSTR('Cadeira', 'a') "Ocorrência" FROM DUAL;



# 1.2.2.3 **LENGTH**

LENGTH (col | string)

Devolve o comprimento da string.

# Exemplo:

Apresente o comprimento da palavra 'Cadeira'.

SELECT LENGTH('Cadeira') "Comprimento" FROM DUAL;



EIBD: funções Oracle Página 13 de 31

# 1.3 Funções de Data

# 1.3.1 ADD MONTHS

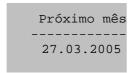
```
ADD_MONTHS (data, n)
```

Devolve uma data, resultado da soma de n meses com data. O valor n tem que ser inteiro mas pode ser negativo.

### Exemplo:

Apresente o mês após a data actual.

```
SELECT ADD_MONTHS(sysdate, 1) "Próximo mês" FROM DUAL;
```



# 1.3.2 LAST DAY

```
LAST_DAY (data)
```

Devolve a data do último dia do mês que contém o dia indicado por data.

### Exemplo:

Apresente o último dia do mês.

```
SELECT LAST_DAY(sysdate) "Último dia" FROM DUAL;
```



# 1.3.3 MONTHS BETWEEN

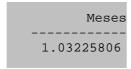
```
MONTHS_BETWEEN (data1, data2)
```

Devolve um número fraccionário, maior, menor ou igual a zero que representa a diferença em meses entre a datal e a datal. Para efeitos da parte fraccionária é considerado que um mês tem 31 dias.

# Exemplo:

Apresente o número de meses entre duas datas.

```
SELECT MONTHS_BETWEEN (TO_DATE ('02-02-1995', 'DD-MM-YYYY'), TO_DATE('01-01-1995', 'DD-MM-YYYY')) "Meses" FROM DUAL;
```



# 1.3.4 NEW\_TIME

```
NEW_TIME (data, fuso1, fuso2)
```

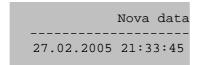
Permite calcular a diferença horária entre dois fusos horários. Consultar os manuais do *Oracle* para mais informações sobre os fusos horários.

EIBD: funções Oracle Página 14 de 31

# Exemplo:

Apresente a diferença entre dois fusos horários.

SELECT NEW\_TIME(sysdate, 'AST', 'PST') "Nova data"
FROM DUAL;



# 1.3.5 NEXT\_DAY

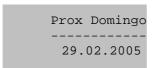
NEXT\_DAY (data, dia\_semana)

Relativamente a data devolve uma nova data que corresponde ao dia da semana (dia\_semana) que vem após data. Em Português valores válidos para dia da semana são: segunda, terça, quarta, quinta, sexta, sábado e domingo. Em Inglês são: sun, mon, tue, wed, thu, fri, sat (também se podem usar nomes completos). Também podemse usar os números de 1(Domingo) a 7 (Sábado).

# Exemplo:

Apresente a data do próximo Domingo.

SELECT NEXT\_DAY(sysdate, 'sun') "Prox Domingo" FROM DUAL;



# 1.3.6 NLS\_DATE\_FORMAT

O *Oracle* representa as datas segundo o formato definido em NLS\_DATE\_FORMAT. Pode-se alterar o seu valor através do comando ALTER SESSION.

Alguns dos comandos válidos do NLS\_DATE\_FORMAT são:

DDD	Dia do ano (número)
DD	Dia do mês (número)
DAY	Dia da semana (extenso)
MM	Número do mês
MON	Nome abreviado do mês
MONTH	Nome por extenso do mês
YYYY	Representar o ano com 4, 3, 2 ou dígitos respectivamente
YYY	
YY	
Y	
НН	Hora do dia (0-12)
нн12	
нн24	Hora do dia (0-24)
MI	Minutos
·	·

EIBD: funções Oracle Página 15 de 31

SS	Segundos
SSSSS	Segundos depois da meia-noite
- / , . ; : "texto"	Pontuação ou texto entre aspas

## Exemplo:

Alterar o formato das datas:

```
ALTER SESSION

SET NLS_DATE_FORMAT= ' "Data:" YYYY-MON-DD HH24:MI:SS';

SELECT SYSDATE FROM DUAL;
```

```
SYSDATE
-----
Data: 2005-FEV-27 21:53:35
```

O comando ALTER SESSION permite modificar muitas outras opções NLS (*National Language Support* – tais como a língua, o território, a moeda, etc.) e ainda outro tipo de opções relacionadas com a ligação à base de dados. Consulte os manuais do *Oracle* para saber mais sobre o comando ALTER SESSION. Para poder ser usado o utilizador precisa ter atribuído o privilégio de sistema com o mesmo nome.

# 1.3.7 **ROUND** (data)

```
ROUND (data, ['DAY' | 'MONTH' | 'YEAR' | 'outro'])
```

Arredonda uma data. A precisão do arredondamento é feita ao dia, semana, mês ou ano.

Para arredondar ao dia não se coloca qualquer tipo de formato. Os valores de tempo antes do meio-dia são arredondados para as zero horas desse dia. Os valores de tempo após o meio-dia são arredondados para o dia seguinte.

Utiliza-se: 'DAY' para arredondar ao dia da semana (para a segunda-feira mais perto); 'MONTH' Para arredondar ao mês; 'YEAR' para arredondar ao ano.

O meio da semana é quarta-feira ao meio-dia. Tudo o que estiver compreendido entre uma segunda-feira e o meio-dia da quarta-feira da mesma semana é arredondado para as zero horas dessa segunda-feira. Tudo o que estiver depois do meio dia de quarta-feira da mesma semana é arredondado para as zero horas da segunda-feira da semana seguinte.

Em termos de meses, considera-se que a metade é à meia noite entre os dias 15 e 16 independentemente do tamanho dos meses. Em termos de anos, considera-se metade a meia-noite entre o dia 30 de Junho e o dia 01 de Julho muito embora a segunda "metade" seja constituída por 184 dias e a primeira por 181 dias( ou 182 em anos bissextos). Em qualquer dos casos, o valor das horas, minutos e segundos fica igual a zero.

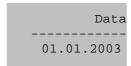
Para outras opções de arredondamento, consulte os manuais da *Oracle*.

#### Exemplo:

Apresente a data arredondada para o primeiro dia do próximo ano:

EIBD: funções Oracle Página 16 de 31

SELECT ROUND(TO\_DATE('03.10.2002', 'DD.MM.YYYY'), 'YEAR')
"Data" FROM DUAL;



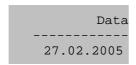
#### 1.3.8 SYSDATE

Função sem argumentos que devolve a data e hora do servidor.

# Exemplo:

Devolve a data e hora do servidor.

SELECT SYSDATE "Data" FROM DUAL;



# 1.3.9 **TRUNC** (data)

```
TRUNC (data,['DAY' | 'MONTH' | 'YEAR' | 'outro'])
```

Arredonda a data para as zero horas do dia actual (se não se usar o segundo argumento), ou para o princípio da semana actual (se o segundo argumento for 'DAY'), ou para o princípio do mês actual (se o segundo argumento for 'MONTH'), ou para o princípio do ano actual (se o segundo argumento for 'YEAR'). Em qualquer dos casos, o valor de horas, minutos e segundos fica igual a zero. Consultar os manuais do *Oracle* para mais informações.

### Exemplo:

Apresente a data truncada para o primeiro dia do corrente ano.

```
SELECT TRUNC(TO_DATE('03.10.2002'), 'YEAR') "Data"
FROM DUAL;
```

Data -----01.01.2002

EIBD: funções Oracle Página 17 de 31

# 1.4 Funções de Conversão

# 1.4.1 ASCIISTR

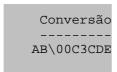
```
ASCIISTR (col | string)
```

Transforma a *string* de texto de entrada para uma *string* ASCII. O valor retornado contém apenas caracteres que aparecem no SQL.

## Exemplo:

Apresente a conversão de 'ABÃCDE'.

```
SELECT ASCIISTR('ABACDE') "Conversão" FROM DUAL;
```



# 1.4.2 **BIN\_TO\_NUM**

```
BIN_TO_NUM (expr1,[expr2,...])
```

Converte um vector de bits num número equivalente. Cada argumento para esta função representa um bit. Cada expr deve conter 0 ou 1.

# Exemplo:

Apresente a conversão do valor binário 1010 para número.

```
SELECT BIN_TO_NUM(1, 0, 1, 0) "Conversão" FROM DUAL;
```



# 1.4.3 CAST

```
CAST (valor AS tipo_dados])
```

Converte um valor especificado para um tipo de dados pré-definido ou criado pelo utilizador. Consulte os manuais do *Oracle* para mais informações.

# Exemplo:

Apresente a conversão da data 1997-10-22.

```
SELECT CAST('22-10-97' AS DATE) "Conversão" FROM DUAL;
```



# 1.4.4 COMPOSE

```
COMPOSE('string')
```

Recebe uma *string* em qualquer tipo e retorna um conjunto de caracteres *Unicode* na forma normalizada.

EIBD: funções Oracle Página 18 de 31

# Exemplo:

Apresente Ö através da função COMPOSE.

```
SELECT COMPOSE('O' || UNISTR('\0308')) "COMPOSE"
FROM DUAL;
```



### 1.4.5 CONVERT

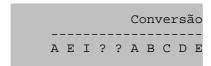
CONVERT(char, destino, origem)

Converte uma *string* de caracteres de um determinado formato para outro. O primeiro parâmetro é o valor a ser convertido que pode ser de qualquer tipo. O segundo parâmetro é o formato para o qual se deseja converter. O último parâmetro é o formato do valor que quer ser convertido. Pode consultar os formatos existentes nos manuais do *Oracle*.

# Exemplo:

Apresente a conversão de um texto em Latin-1para ASCII.

```
SELECT CONVERT('Ä Ê Í Õ Ø A B C D E ', 'US7ASCII', 
'WE8ISO8859P1') "Conversão" FROM DUAL;
```



# 1.4.6 DECOMPOSE

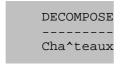
DECOMPOSE('string')

Esta função é válida apenas para caracteres *Unicode*. Recebe uma *string* em qualquer tipo e retorna um conjunto de caracteres *Unicode*, após ter feito a decomposição.

## Exemplo:

Apresente a decomposição da string 'Châteaux'.

```
SELECT DECOMPOSE('Châteaux') "DECOMPOSE" FROM DUAL;
```



# 1.4.7 TO\_CHAR (caracter)

```
TO_CHAR (col | valor)
```

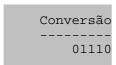
Transforma o parâmetro de entrada para um valor do tipo string.

#### Exemplo:

Apresente a conversão de '01110'.

EIBD: funções Oracle Página 19 de 31

SELECT TO\_CHAR('01110') "Conversão" FROM DUAL;



# 1.4.8 **TO\_CHAR** (data)

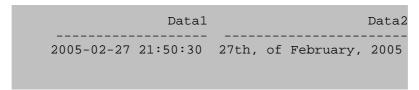
```
TO_CHAR (d [, fmt[, 'nlsparams']])
```

Permite transformar um valor do tipo data num valor do tipo string. A string fmt define o tipo de transformação e é construída através dos campos vistos em NLS\_DATE\_FORMAT. Ver significado dos 'nlsparams' na documentação do Oracle.

# Exemplo:

Apresente a data do servidor transformada num valor do tipo string.

SELECT TO\_CHAR(sysdate, 'yyyy-mm-dd hh24:mi:ss') "Data1",
TO\_CHAR(sysdate, 'fmddth, "of" Month, yyyy') FROM DUAL;



# 1.4.9 TO\_CHAR (número)

TO\_CHAR (n [, fmt[, 'nlsparams']])

A string fmt define como é feita a transformação de números para caracteres. O significado dos 'nlsparams' deve ser consultado na documentação *Oracle*. A string de formatação fmt pode ter os seguintes elementos e respectivos significados:

Símbolo	Significado		
9	Um símbolo 9 para cada algarismo significativo a representar antes ou depois da vírgula (note-se que nas <i>strings</i> de formatação a vírgula é representada por um ponto por herança da numeração anglo-saxónica). Se existirem mais 9s que algarismos a representar só são representados os dígitos que existem.		
0	Faz o mesmo que o 9 mas se o número não tiver tantos algarismos como definido em fmt é incluído o algarismo 0 por cada um que não existe.		
В	Se o número for 0 apresenta apenas um espaço.		
S	Coloca-se antes ou depois da sequência de 0s e/ou 9s e indica onde deve aparecer o sinal do número. Se S não for indicado aparece o sinal apenas para os números negativos e nos positivos aparece um espaço.		
L	Quando usado, o valor aparece como uma representação monetária. É incluído o símbolo (\$) ou outro dependendo dos parâmetros NLS. Pode ser colocado antes ou depois da sequência de 0s e/ou 9s.		
Fm	Se usado, os espaços a mais são removidos.		

EIBD: funções Oracle Página 20 de 31

# Exemplos:

Ε

Número	'fmt'	Resultado
-1234567890	999999999S	`1234567890-'
0	99.99	`0.00′
+0.1	99.99	` .10′
-0.2	99.99	·20′
0	90.99	` 0.00′
+0.1	90.99	` .10′
-0.2	90.99	` -0.20 <i>'</i>
0	9999	` 0'
1	9999	` 1'
0	в9999	V 1
1	в9999	` 1'
0	в90.99	V /
+123.456	999.999	` 123.456′
-123.456	999.999	`-123.456 <i>'</i>
+123.456	FM999.009	`123.456′
+123.456	9.9EEEE	` 1.2E+02'
+1E+123	9.9EEEE	` 1.0E+123'
+123.456	FM9.9EEEE	`1.23E+02 <i>'</i>
+123.45	FM999.009	`123.45 <i>'</i>
+123.0	FM999.009	`123.00 <i>'</i>
+123.45	L999.99	`\$123.45 <i>'</i>
+123.45	FML99.99	`\$123.45′
+1234567890	9999999999S	`1234567890+'

# 1.4.10 TO\_CLOB

TO\_CLOB (col | char)

Converte strings de caracteres para CLOBs.

# Exemplo:

Converta os dados do tipo NCLOB da tabela  $pm.print\_media$  para o tipo CLOB e insira-os numa coluna do tipo CLOB, substituindo os dados na coluna.

UPDATE print\_media SET ad\_finaltext=TO\_CLOB(ad\_fltextn);

EIBD: funções Oracle Página 21 de 31

# 1.4.11 TO DATE

```
TO_DATE (char [, fmt[, 'nlsparams']])
```

Transforma uma string numa data.

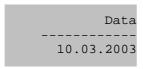
A função TO\_CHAR faz a operação inversa, ou seja, transforma uma data numa *string*. O formato da *string* de entrada deve ser especificado em *fmt*.

Ver também NLS\_DATE\_FORMAT, ALTER SESSION e a documentação do *Oracle*. Esta função é usada frequentemente para inserir valores de datas.

# Exemplo:

Apresente a string '2003/03/10' no formato data.

```
SELECT TO_DATE('2003/03/10', 'yyyy/mm/dd') "Data"
FROM DUAL;
```



# 1.4.12 TO\_LOB

```
TO_LOB (long_column)
```

Converte valores do tipo LONG e LONG RAW para valores do tipo LOB.

#### Exemplo:

Converter dados para valores do tipo LOB.

```
CREATE TABLE new_print_media(
    Product_id NUMBER(6),
    ad_id NUMBER(6),
    press_release CLOB);

INSERT INTO new_print_media
    (SELECT p.product, p.ad_id, TO_LOB(p.press_release)
    FROM print_media p);
```

### **1.4.13 TO NUMBER**

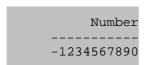
```
TO_NUMBER (char [, fmt[, 'nlsparams']])
```

Transforma uma *string* numa número exactamente da mesma maneira que TO\_CHAR (para converter números) transforma um número numa *string*. Ver a documentação do *Oracle*.

### Exemplo:

Apresente a string '1234567890-' transformada num número.

```
SELECT TO_NUMBER('1234567890-','99999999999') "Number"
FROM DUAL;
```



EIBD: funções Oracle Página 22 de 31

# 1.4.14 UNISTR

UNISTR ('string')

Recebe uma *string* com um conjunto de caracteres ASCII e valores *Unicode* e converte-a para os caracteres da língua especificada no *Oracle*.

# Exemplo:

Apresente uma string convertida.

SELECT UNISTR('abc\00e5\00f1\00f6') "Conversão"
FROM DUAL;



EIBD: funções Oracle Página 23 de 31

# 1.5 Funções Miscelânea

### 1.5.1 BFILENAME

```
BFILENAME(directoria, nome_ficheiro)
```

Devolve um ponteiro para um ficheiro LOB, armazenado no sistema de ficheiros do servidor *Oracle*.

# Exemplo:

Insira uma linha na tabela *pm.print\_media*. Use esta função, para identificar um ficheiro binário no sistema de ficheiros do servidor.

```
CREATE DIRECTORY media_dir AS '/demo/schema/
product_media';
INSERT INTO print_media (product_id, ad_id, ad_graphic)
VALUES(300, 31001, BFILENAME('MEDIA_DIR',
'media_comp_ad.gif'));
```

# 1.5.2 COALESCE

```
COALESCE(expr1 [, expr2,...])
```

Retorna a primeira expressão não nula de uma lista de expressões. Esta função é uma generalização da função NVL.

```
COALESCE (expr1, expr2)
É equivalente a:
    CASE WHEN expr1 IS NOT NULL THEN expr1 ELSE expr2 END
COALESCE(expr1, expr2, ..., exprn), para n>3
```

### É equivalente a:

```
CASE WHEN expr1 IS NOT NULL THEN expr1 
ELSE COALESCE (expr2, ..., exprn) END
```

#### Exemplo:

Apresente uma listagem de produtos e o respectivo valor da venda. O valor da venda é dado com 10% de desconto se o valor list\_price não for nulo. Se não houver list\_price, o preço de venda é o min\_price. Se não houver min\_price, então o preço de venda é 5.

```
SELECT product_id, list_price, min_price,

COALESCE (0.9*list_price, min_price,5) "Venda"

FROM product_information

WHERE supplier id=100250;
```

PRODUCT_ID	LIST_PRICE	MIN_PRICE	VENDA
 			2382
	850	731	765
3355			5.
1770		73	73
1769	48		43.2

EIBD: funções Oracle Página 24 de 31

# 1.5.3 Expressão CASE

A expressão CASE permite o uso de IF...THEN...ELSE dentro de instruções SQL e pode ser usada de duas formas:

### > CASE simples:

```
CASE expr
WHEN expr_compara THEN expr_resultado
[WHEN expr_compara2 THEN expr_resultado2]
...
[ELSE else_resultado]
END
```

Procura a primeira expr\_compara que seja igual a expr devolvendo o expr\_resultado associado. Se nenhuma das expressões for igual a expr devolve else\_resultado ou *NULL* caso o ELSE não exista.

## Exemplo:

Apresente o último nome do cliente e caso o limite do crédito dos clientes seja igual a 100 escreva 'Baixo', se for igual a 5000 escreva 'Alto', caso contrário escreva 'Médio'.

```
SELECT cust_last_name, CASE credit_limit
WHEN 100 THEN 'Baixo'
WHEN 5000 THEN 'Alto'
ELSE 'Médio' END
```

FROM customers;

CUST_LAST_NAME	CASECR
Bogart Nolte	 Médio Médio 

#### > CASE pesquisa:

```
CASE
WHEN condicao THEN expr_resultado
[WHEN condicao2 THEN expr_resultado2]
...
[ELSE else_resultado]
END
```

Testa as condições uma a uma, até encontrar a primeira condição verdadeira e devolve expr\_resultado associado à respectiva condição. Se nenhuma das condições for verdadeira devolve else\_resultado ou *NULL* caso o ELSE não exista.

### Exemplo:

Apresente a média dos salários dos empregados, usando 2000 como o salário mais baixo possível.

2634.32429

EIBD: funções Oracle Página 25 de 31

### 1.5.4 **DECODE**

```
DECODE(expr, compara1, resultado1[, compara2, resultado2, ...],
valor omissao)
```

Permite fazer testes semelhantes aos IFs de outras linguagens. O primeiro parâmetro é comparado com comparal. Se for igual é devolvido o valor do resultadol. Caso contrário é comparado com compara2 e assim sucessivamente. Se não for igual a nenhum dos comparas então é devolvido o valor de valor omissao.

## Exemplo:

Apresente a função dos empregados da tabela emp e respectiva codificação.

```
SELECT job, DECODE(job, 'PRESIDENT', 'CHEFE', 'MANAGER',
'ADMINISTRADOR', 'INDEFINIDO') "Decode"
FROM emp;
```

FUNCAO	DECODE
PRESIDENT MANAGER	CHEFE ADMINISTRADOR
SECRETARY	INDEFINIDO

# 1.5.5 **EMPTY\_[B | C]LOB**

```
EMPTY_BLOB()
EMPTY CLOB()
```

Criam um ponteiro que serve para inicializar uma variável LOB.

# Exemplo:

Inicialize uma variável do tipo LOB.

```
UPDATE print_media SET ad_photo=EMPTY_BLOB();
```

# 1.5.6 GREATEST

```
GEATEST(expr [, expr] ...)
```

Devolve o maior dos parâmetros de entrada. Se, para a comparação for necessário, converte todos os parâmetros para o tipo de dados usado no primeiro parâmetro.

#### Exemplo:

Apresente o maior dos 4 elementos 12, 5, 80 e 25.

```
SELECT GREATEST (12, 5, 80, 25) "Greatest" FROM DUAL;
```



# 1.5.7 **LEAST**

```
LEAST(expr [, expr] ...)
```

Devolve o menor dos parâmetros de entrada. Se, para a comparação for necessário, converte todos os parâmetros para o tipo de dados usado no primeiro parâmetro.

#### Exemplo:

EIBD: funções Oracle Página 26 de 31

Apresente o menor dos 4 elementos 12, 5, 80 e 25.

SELECT LEAST (12, 5, 80, 25) "Least" FROM DUAL;



#### 1.5.8 **NULLIF**

NULLIF(expr1 , expr2)

Compara a expr1 com a expr2. Se forem iguais, retorna null. Caso contrário retorna expr1.

# Exemplo:

Apresente a profissão dos empregados caso estes tenham mudado de profissão desde que foram contratados, como indicado pelo *job\_id* da tabela *job\_history* se for diferente do *job\_id* da tabela *employee*.

```
SELECT e.last_name, NULLIF (e.job_id, j.job_id) "Job ID
antigo" FROM employees e, job_history j
WHERE e.employee_id=j.employee_id
ORDER BY last_name;
```

LAST_NAME	Job ID antigo
De Haan	AD_VP
Taylor Whalen	AD_ASST

# 1.5.9 NVL

NVL(expr1, expr2)

Devolve expr2 se expr1 tiver valor nulo. Caso contrário devolve expr1.

### Exemplo:

Apresente uma lista dos nomes dos empregados e as comissões de cada um, substituindo a comissão por 'Não aplicável' caso o empregado não tenha comissão.

```
SELECT Ename "Nome", NVL (TO_CHAR(comm), 'Não aplicável') "Comissão" FROM emp;
```

NOME	COMISSÃO
SMITH ALLEN WARD	Não aplicável 300 500
• • •	• • •

1.5.10 NVL2

NVL2(expr1, expr2, expr3)

Permite determinar o valor retornado caso a expressão especificada seja ou não nula. Devolve expr2 se expr1 não tiver valor nulo. Caso contrário devolve expr3.

Exemplo:

EIBD: funções Oracle Página 27 de 31

Apresente uma lista dos nomes dos empregados e o salário com comissão ou sem comissão casos estes tenham ou não comissão.

SELECT Ename "Nome", sal "Salario", NVL2 (comm, sal\* (1+comm), sal) "Resultado" FROM emp;

NOME	SALARIO	RESULTADO
SMITH ALLEN WARD	800 1600 1250	800 481600 626250
•••	•••	•••

# 1.5.11 UID

Devolve o número do utilizador.

Exemplo:

SELECT UID FROM DUAL;

# 1.5.12 USER

Devolve o nome do utilizador.

Exemplo:

SELECT USER FROM DUAL;

# 1.5.13 **USERENV**

USERENV(option)

Permite verificar os valores das variáveis de sessão como 'LANG', 'LANGUAGE', 'TERMINAL', 'SESSIONID', 'CLIENT\_INFO' e outras. Consulte os manuais da *Oracle* para mais informações.

### Exemplo:

Apresente o valor da variável de sessão 'LANGUAGE'.

SELECT USERENV('LANGUAGE') "Linguagem" FROM DUAL;

# 1.5.14 **VSIZE**

VSIZE(expr)

Devolve o tamanho em bytes de uma expressão.

# Exemplo:

Apresente o número de bytes dos nome dos empregados do departamento 20.

SELECT ename, VSIZE(ename) "BYTES" FROM emp
WHERE deptno=10;

ENAME	BYTES
CLARK	5
KING	4
MILLER	6

EIBD: funções Oracle Página 28 de 31

# 2 Funções de grupo

As funções de grupo têm as seguintes características:

- Devolvem apenas um valor para um conjunto de linhas.
- Podem ser usadas nas cláusulas SELECT e HAVING.
- Cada um dos valores devolvidos é calculado para o conjunto de registos definidos pela cláusula GROUP BY.
- Sem GROUP BY os valores são calculados para a tabela toda.
- Todas as funções de grupo ignoram os nulos excepto o COUNT (\*).
- A cláusula DISTINCT permite eliminar os repetidos.
- Não se pode misturar funções de grupo com funções que não são de grupo.
- HAVING apenas para restrições de grupo, WHERE apenas para de linha.

# 2.1 **AVG**

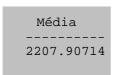
```
AVG ([DISTINCT | ALL] col | num)
```

Devolve a média de valores que essa expressão numérica representa. Apesar dos valores da coluna serem apenas inteiros, o valor da média pode ser um número fraccionário.

# Exemplo:

Apresente a média de salários da tabela emp.

```
SELECT AVG(sal) "Média" FROM emp;
```



# **2.2 COUNT**

```
COUNT ([DISTINCT | ALL] valor | *)
```

Devolve o número de registos da tabela que correspondem a uma expressão não nula.

O caso especial COUNT(\*) devolve o número de registos desse grupo, mesmo que existam valores nulos.

### Exemplo:

Apresente o número total de empregados da tabela emp.

```
SELECT COUNT(*) "Total" FROM emp;
```



EIBD: funções Oracle Página 29 de 31

# 2.3 MAX

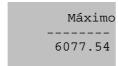
```
MAX ([DISTINCT | ALL] valor)
```

Devolve o maior valor de um conjunto de valores.

Exemplo:

Apresente o salário mais elevado da tabela emp.

```
SELECT MAX(sal) "Máximo" FROM emp;
```



# 2.4 MIN

```
MIN ([DISTINCT | ALL] valor)
```

Devolve o menor valores de um conjunto de valores.

Exemplo:

Apresente o salário mais baixo da tabela *emp*.

```
SELECT MIN(sal) "Mínimo" FROM emp;
```



# 2.5 STDDEV

```
STDDEV ([DISTINCT | ALL] valor)
```

Devolve o desvio padrão que essa expressão representa.

Exemplo:

Apresente o desvio padrão dos salários da tabela emp.

```
SELECT STDDEV(sal) "Desvio Padrão" FROM emp;
```

```
Desvio padrão
-----
1406.42897
```

# 2.6 SUM

```
SUM ([DISTINCT | ALL] valor)
```

Devolve a soma de valores que essa expressão representa.

Exemplo:

Apresente a soma de salários da tabela *emp*.

Soma -----30910.7

EIBD: funções Oracle Página 30 de 31

# 2.7 VARIANCE

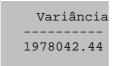
```
VARIANCE ([DISTINCT | ALL] valor)
```

Devolve a variância de uma expressão. Consulte o manual da *Oracle* para mais informações.

# Exemplo:

Apresente a variância do salário dos empregados da tabela emp.

```
SELECT VARIANCE(sal) "Variância" FROM emp;
```



# 2.8 Extensões

Existem extensões à cláusula GROUP BY, que não sendo funções podem ser usadas para obter super agregações de linhas. É o caso das extensões ROLLUP e CUBE.

## 2.8.1 ROLLUP

Produz uma nova agregação por cada grupo expressando os valores totais de um grupo. Ver GROUPING.

#### 2.8.2 CUBE

Idêntica à ROLLUP, mas cria uma nova agregação com os subtotais de cada uma das expressões do grupo. Ver GROUPING.

### 2.8.3 GROUPING

```
GROUPING (expr)
```

As extensões da cláusula GROUP BY, tais como o ROLLUP e o CUBE, produzem valores aos quais não são atribuídos nomes. Com o GROUPING há a possibilidade de atribuir um nome (*label*) a valores nas linhas super agregadas.

# Exemplo:

Apresente a soma dos salários dos empregados por profissão.

Job	Salario
ANALYST	6000
CLERK	4150
MANAGER	8275
PRESIDENT	5000
SALESMAN	5600
all_jobs	29025

EIBD: funções Oracle Página 31 de 31