Numerikus függvények

ABS(n)
ACOS(n)
ASIN(n)
ATAN(n), ATAN2(n, m) ATAN2(n, m) = ATAN(n/m)
BITAND(positiv_int, positiv_int2) bitenkénti és művelet
CEIL(n) felső egészrész
COS(n)
COSH(n)
EXP(n) e az n-ediken
FLOOR(n) egészrész
LN(n)
LOG(m, n)
MOD(m, n) n=0 esetén m-et ad vissza
POWER(m, n) m az n-ediken
ROUND(n [, int]) kerekítés. int lehet <0 is, alapértelmezése = 0
SIGN(n) előjel
SIN(n)
SINH(n)
SQRT(n) négyzetgyök
TAN(n)
TANH(n)
TRUNC(n [, int]) csonkolás. int lehet <0 is, alapértelmezése = 0
WIDTH_BUCKET(kif, min_érték, max_érték, kosarak_száma) azonos szélességű hisztogram esetén melyik kosárba esne a kifejezés

Karakterkezelő függvények

ASCII(str) az első karakter ascii kódja
CHR(n) az n kódú karakter az adott kódkészletben
CONCAT(str, str)
LOWER(str)
UPPER(str)
INITCAP(str) szavak kezdőbetűi nagybetűssé alakítva
LENGTH(str) karakterlánc hossza
SUBSTR(str, pozíció [,hossz]) részkarakterlánc. pozíció < 0 esetén hátulról számol
INSTR(str, str [,pozíció] [,előfordulás]) keresett részkarakterlánc kezdete
LPAD(str, hossz [,str2]) balról adott hosszúságúra való kiegészítés (default str2 = ' ')
RPAD(str, hossz [,str2]) jobbról adott hosszúságúra való kiegészítés
LTRIM(str [,str2]) str2-beli karakterek eltávolítása str bal oldaláról (default str2 = ' ')
RTRIM(str [,str2]) str2-beli karakterek eltávolítása str jobb oldaláról (default str2 = ' ')
TRIM([LEADING | TRAILING | BOTH str] FROM str)
NLS_LOWER
NLS_UPPER
NLS_INITCAP
NLS_SORT adott nyelven történő rendezéshez
REPLACE(str, mit [,mire]) str-beli karakterláncok lecserélése (vagy kivágása)
TRANSLATE(str, 'input_karakterek', 'csere_karakterek')

Dátumkezelő függvények

SYSDATE	rendszerdátum
ADD_MONTHS(d, n)	n hónap hozzáadása d-hez
MONTHS_BETWEEN(d, d)	az eltelt hónapok száma
LAST_DAY(d)	az adott hónap utolsó napja
NEXT_DAY(d, str)	a legközelebbi adott nevű nap d után
ROUND(d, [, formátum])	kerekítés
TRUNC(d, [, formátum])	csonkolás

Dátum és időkezelés (9-estől)

Új típus: TIMESTAMP

```
TIMESTAMP[(törtmásodpercek pontossága)]    -- alapértelmezés=6
CREATE TABLE  T(ido TIMESTAMP);
INSERT INTO  T  VALUES(TIMESTAMP '1997-01-31 09:26:50.124');

TIMESTAMP[(törtmásodpercek pontossága)] WITH TIME ZONE
CREATE TABLE  T(ido TIMESTAMP WITH TIME ZONE);
INSERT INTO  T  VALUES(TIMESTAMP '1997-01-31 09:26:50.124 +02:00');

TIMESTAMP[(törtmásodpercek pontossága)] WITH LOCAL TIME ZONE
Az adatbázis időzónájában tárolódik, és a kliens lokális időzónájában
jelenik meg az adat. Nincs tárolva az időzóna eltolás.
```

Session időzónájának beállítása: ALTER SESSION SET **TIME_ZONE** = '+2:0';
Új függvények:
Adatbázis ill. session időzónája: **DBTIMEZONE**, **SESSIONTIMEZONE**
Aktuális dátum a session időzónájában: **CURRENT_DATE** (DATE típus)
Aktuális dátum időzónával együtt: **CURRENT_TIMESTAMP** (TSTZ típus)
Akt. dátum időzónával a session zónájában: **LOCALTIMESTAMP** (TSTZ típus)
EXTRACT([year] [month] [day] [hour] [minute] [second] [timezone_hour] … FROM <dátum>)
TS konvertálása TSTZ-vé: **FROM_TZ**(TIMESTAMP '2004-04-11 08:12:14', '2:00')
TS-re ill. TSTZ-re: **TO_TIMESTAMP**('2004-04-11 08:12:14', 'YYYY-MM-DD HH:MI:SS')
TO_TIMESTAMP_TZ('2004-04-11 08:12:14 +2:00', 'YYYY-MM-DD HH:MI:SS TZH:TZM')
Időzóna eltolódás: **TZ_OFFSET**('Europe/Budapest') -> +02:00

Példák intervallum (INTERVAL) típusokra

```
CREATE TABLE time_table (
   start_time      TIMESTAMP,
   duration_1      INTERVAL DAY (6) TO SECOND (5),
   duration_2      INTERVAL YEAR TO MONTH);

-- INTERVAL DAY [(day_precision)] TO SECOND
[(fractional_seconds_precision)]
-- INTERVAL YEAR [(year_precision)] TO MONTH

INTERVAL '4 5:12:10.222' DAY TO SECOND(3)
```

```
INTERVAL '123-2' YEAR(3) TO MONTH


SELECT last_name, EXTRACT(YEAR FROM (SYSDATE - hire_date) YEAR TO MONTH )
|| ' years '
   || EXTRACT(MONTH FROM (SYSDATE - hire_date) YEAR TO MONTH ) || ' months'
FROM hr.employees;

SELECT order_id, EXTRACT(DAY FROM (SYSDATE - order_date) DAY TO SECOND ) ||
' days '
   || EXTRACT(HOUR FROM (SYSDATE - order_date) DAY TO SECOND ) || ' hours '
   || EXTRACT(MINUTE FROM (SYSDATE - order_date) DAY TO SECOND ) || '
minutes '
   || EXTRACT(SECOND FROM (SYSDATE - order_date) DAY TO SECOND ) || '
seconds'
FROM oe.orders
```

Konverziós függvények

TO_CHAR(d [, fmt [, nlsparam]])
TO_CHAR(n [, fmt [, nlsparam]])
TO_NUMBER(str [, fmt [, nlsparam]])
TO_DATE(str [, fmt [, nlsparam]])
CHARTOROWID(str)
ROWIDTOCHAR(rowid)

További függvények

NVL(expr1, expr2) ha expr1 is NULL -> expr2 egyébként -> expr1
NVL2(expr1, expr2, expr3) ha expr1 is NULL -> expr3 egyébként -> expr2
NULLIF(expr1, expr2)
CASE WHEN expr1 = expr 2 THEN NULL ELSE expr1 END

COALESCE(expr1, expr2, …) az első nem NULL értéket adja vissza
DECODE(expr, search1, result1 [, search2, result2,...,] [, default])

CASE WHEN feltétel1 THEN result1
[WHEN feltétel2 THEN result2 …
ELSE default]
END

CASE expr WHEN search1 THEN result1
[WHEN search2 THEN result2 …
ELSE default]
END

USER az aktuális felhasználó neve
UID az aktuális felhasználó rendszerbeli azonosító kódja
GREATEST(expr_list) legnagyobb elem
LEAST(expr_list) legkisebb elem

Reguláris kifejezések kezelése (10.1-től)

'*' nulla vagy több előfordulás, '**+**' 1 vagy több előfordulás, '**?**' 0 vagy 1 előfordulás
'^' sor kezdete, '**$**' sor vége, '**.**' tetszőleges karakter, '**[]**' bármelyik karakter a zárójelben,
'[: :]' karakterosztályok pl. [:alpha:], [:digit:], [:alnum:], [:blank:], [:punct:] … stb.
'()' csoportosító kifejezés, **'|'** vagylagos operátor
{m,} {m} {m, n} illeszkedések száma legalább m, pontosan m, m és n közötti
'\n' visszahivatkozó kifejezés az n-edik zárójelezett kifejezésre

```
REGEXP_SUBSTR(forrás, minta, pozíció, előfordulás,
illeszkedési param)
```

Visszaadja a keresett mintát (vagy NULL-t) a forrás megadott részében (poz) keresve. Az illeszkedési paraméter jelezheti pl. hogy case sensitive módon keresünk-e vagy nem ('c', 'i').

```
-- Az 'a' betű és a két utána következõ karakter
SELECT regexp_substr('123:_*+abc:_*+123:-*+ABC', '[a]..') FROM
dual;
-- abc

-- Az 'a' betű és a két utána következõ karakter második elõfordulása
SELECT regexp_substr('123:_*+abc:_*+123:-*+ABC', '[a]..', 1,
2) FROM dual;
-- NULL

-- Az 'a' betű és a két utána következõ karakter a forrás 9. karakterétől kezdődő részben
SELECT regexp_substr('123:_*+abc:_*+123:-*+ABC', '[a]..', 9)
FROM dual;
-- NULL

-- Az 'a' betű és a két utána következõ karakter második elõfordulása, nem case sensivite
esetben
SELECT regexp_substr('123:_*+abc:_*+123:-*+ABC', '[a]..', 1,
2, 'i') FROM dual;
-- ABC

-- A két '_' közötti rész
SELECT regexp_substr('123:_*+abc:_*+123:-*+ABC', '_[^_]*_')
FROM dual;
-- _*+abc:_

-- 3 számjegy egymás után
SELECT regexp_substr('123:_*+123:-*+ABC', '([[:digit:]]{3})')
FROM dual;
-- 123

-- 1 vagy 2
SELECT regexp_substr('123:_*+123:-*+ABC', '(1|2)') FROM dual;
-- 1
```

```
REGEXP_INSTR(forrás, minta, poz, előf, visszatérési opc,
illeszkedési param)
```
Visszaadja a megtalált minta első karakterének pozícióját (opc=0), vagy a minta utáni pozíciót (opc=1). Ha nem talál, 0-t ad vissza.

```
-- A két '_' közötti rész előfordulásának pozíciója
SELECT regexp_instr('123:_*+abc:_*+123:-*+ABC', '_[^_]*_')
FROM dual;
-- 5

-- A minta utáni pozíció
SELECT regexp_instr('123:_*+abc:_*+123:-*+ABC', '_[^_]*_', 1,
1, 1) FROM dual;
-- 13

-- A '*' 0-szori elõfordulást is megenged, az alábbi minta
illeszkedik
SELECT regexp_instr('1223:_*+abc:_*+1223:-*+ABC', '(22)*')
FROM dual;
-- 1

-- A '+' csak >=1-szeri előfordulást enged
SELECT regexp_instr('1223:_*+abc:_*+1223:-*+ABC', '(22)+')
FROM dual
-- 2
```

```
REGEXP_LIKE(forrás, minta, poz, előf, visszatérési opc,
illeszkedési param)
```
Igaz vagy hamis értékkel tér vissza attól függően, hogy talált-e mintát.

```
-- Az 'A' betű után már nincs 3 karakter
SELECT 'X' from dual WHERE regexp_like('123:_*+abc:_*+123:-
*+ABC', '[A]...')
-- No rows selected
```

```
REGEXP_REPLACE(forrás, minta, csere, poz, előf, illeszkedési
param)
```
Ha az illeszkedési param = 0 akkor a minta összes előfordulása le lesz cserélve a csere stringre, ha n>0, akkor csak az n-edik.

```
-- Formázzunk meg egy mobil telefonszámot
SELECT regexp_replace('30-3456789',
'([[:digit:]]{2})-([[:digit:]]{3})',
                      '(\1) \2-') FROM dual
-- (30) 345-6789
```

Példák függvényekre

Függvényhívás	Eredmény
LOWER('SQL Course')	sql course
UPPER('SQL Course')	SQL COURSE
INITCAP('SQL Course')	Sql Course
Karakterkezelő fv-ek	
CONCAT('Good', 'String')	GoodString
SUBSTR('String',1,3)	Str
LENGTH('String')	6
INSTR('String', 'r')	3
LPAD(sal,10,'*')	******5000
Kerekítés, csonkolás	
ROUND(45.926, 2)	45.93
TRUNC(45.926, 2)	45.92
MOD(1600, 300)	100
Dátum fv-ek	
MONTHS_BETWEEN ('01-SEP-95','11-JAN-94')	19.6774194
ADD_MONTHS ('11-JAN-94',6)	'11-JUL-94'
NEXT_DAY ('01-SEP-95','FRIDAY')	'08-SEP-95'
LAST_DAY('01-SEP-95')	'30-SEP-95'
ROUND('25-JUL-95','MONTH')	'01-AUG-95'
ROUND('25-JUL-95','YEAR')	01-JAN-96
TRUNC('25-JUL-95','MONTH')	01-JUL-95
TRUNC('25-JUL-95','YEAR')	01-JAN-95

Dátumkezelő függvények és dátumaritmetika

d + n -> d
d – d -> n (napok száma)

Implicit adatkonverzió

VARCHAR2 vagy CHAR -> NUMBER
VARCHAR2 vagy CHAR -> DATE
NUMBER -> VARCHAR2
DATE -> VARCHAR2

Explicit konverzió

TO_CHAR, TO_NUMBER, TO_DATE

TO_CHAR(*date, 'fmt*') a formátummodell case sensitive

```
YYYY      Évszám 4 számjeggyel
YEAR      Évszám betűkkel leírva
MM        Hónap két számjeggyel
MONTH     Hónap teljes neve
MON       Hónap rövidített neve
WW        A hét sorszáma az évben
W         A hét sorszáma a hónapban
DDD       A nap sorszáma az évben
DD        A nap sorszáma a hónapban
D         A nap sorszáma a héten
DY        A nap nevének 3 betűs rövidítése
DAY       Nap teljes neve
HH24      Az óra két számjeggyel (0-23)
MI        Perc két számjeggyel
SS        Másodperc két számjeggyel
```

HH24:MI:SS AM -> 15:45:32 PM
DD "of" MONTH -> 12 of OCTOBER
ddspth -> fourteenth

select to_char(sysdate, 'month', 'nls_date_language=hungarian') from dual -> **március**
select to_char(sysdate, 'MONTH', 'nls_date_language=hungarian') from dual -> **MÁRCIUS**
select to_char(sysdate, 'YEAR', 'nls_date_language=hungarian') from dual ->
TWO THOUSAND FOUR (!!!)

TO_CHAR(*number, 'fmt*')
9 -> Represents a number
0 -> Forces a zero to be displayed
$ -> Places a floating dollar sign
L -> Uses the floating local currency symbol
. -> Prints a decimal point
, -> Prints a thousand indicator