Functii SQL

# SELECT

***SELECT*** { [ {***DISTINCT*** *|* ***UNIQUE***} *|* ***ALL***] *lista\_campuri* | \*}

***FROM*** [*nume\_schemă*.]*nume\_obiect* ]

[, [*nume\_schemă*.]*nume\_obiect …*]

[***WHERE*** *condiţie\_clauza\_where*]

[***START WITH*** *condiţie\_clauza\_start\_with*

***CONNECT BY*** *condiţie\_clauza\_connect\_by*]

[***GROUP BY*** *expresie* [*, expresie …*]

[***HAVING*** *condiţie\_clauza\_having*] ]

[***ORDER BY*** {*expresie | poziţie*} [, {*expresie | poziţie*} *…*] ]

[***FOR UPDATE***

[***OF*** [ [*nume\_schemă*.]*nume\_obiect*.]*nume\_coloană*

[, [ [*nume\_schemă*.]*nume\_obiect*.]*nume\_coloană*] *…*]

[***NOWAIT*** *|* ***WAIT*** *număr\_întreg*] ];

# Format dată calendaristică

Datele calendaristice pot fi formatate cu ajutorul funcţiei TO\_CHAR(data, format), unde formatul poate fi alcătuit dintr-o combinaţie a următoarelor elemente

|  |  |
| --- | --- |
| **Element** | **Semnificaţie** |
| D | Numărul zilei din săptămâna (duminica=1; luni=2; …sâmbătă=6) |
| DD | Numărul zilei din lună. |
| DDD | Numărul zilei din an. |
| DY | Numele zilei din săptămână, printr-o abreviere de 3 litere (MON, THU etc.) |
| DAY | Numele zilei din săptămână, scris în întregime. |
| MM | Numărul lunii din an. |
| MON | Numele lunii din an, printr-o abreviere de 3 litere (JAN, FEB etc.) |
| MONTH | Numele lunii din an, scris în întregime. |
| Y | Ultima cifră din an |
| YY, YYY, YYYY | Ultimele 2, 3, respectiv 4 cifre din an. |
| YEAR | Anul, scris în litere (ex: two thousand four). |
| HH12, HH24 | Orele din zi, între 0-12, respectiv 0-24. |
| MI | Minutele din oră. |
| SS | Secundele din minut. |
| SSSSS | Secundele trecute de la miezul nopţii. |

# Funcțiile single row

## Funcții de conversie

|  |  |  |
| --- | --- | --- |
| **Funcţie** | **Descriere** | **Exemplu conversie** |
| *TO\_CHAR* | converteşte (sau formatează) un număr sau o dată calendaristică în şir de caractere | *TO\_CHAR(7) = ' 7'*  *TO\_CHAR(-7) = '-7' TO\_CHAR*  *(SYSDATE, 'DD/MM/YYYY') = '*  *18/04/2007'* |
| *TO\_DATE* | converteşte (sau formatează) un număr sau un şir de caractere în  dată calendaristică | *TO\_DATE('18-APR-2007','ddmon-yyyy')* |
| *TO\_NUMBER* | converteşte (sau formatează) un şir de caractere în număr | *TO\_NUMBER ('-25789', 'S99,999') = -25,789* |

## Funcții pentru prelucrarea caracterelor

|  |  |  |
| --- | --- | --- |
| **Funcţie** | **Descriere** | **Exemplu** |
| *LENGTH(string)* | întoarce lungimea şirului de  caractere *string* | *LENGTH(‘Informatica’)=11* |
| *SUBSTR(string, start*  *[,n])* | întoarce subşirul lui *string* care începe pe poziţia *start* şi are lungimea *n*; dacă *n* nu este specificat, subşirul se termină la  sfârşitul lui *string*; | *SUBSTR(‘Informatica’, 1, 4) =*  *‘Info’*  *SUBSTR(‘Informatica’, 6) =*  *‘matica’*  *SUBSTR(‘Informatica’, -5) =*  *‘matica’* (ultimele 5 caractere) |
| *LTRIM(string*  *[,’chars’])* | şterge din stânga şirului *string* orice caracter care apare în *chars*, până la găsirea primului caracter care nu este în *chars*; în cazul în care *chars* nu este specificat, se şterg spaţiile libere din stânga lui *string*; | *LTRIM (' info') = 'info'* |
| *RTRIM(string*  *[,’chars’])* | este similar funcţiei *LTRIM*, cu excepţia faptului că ştergerea se face la dreapta şirului de caractere; | *RTRIM ('infoXXXX', 'X') =*  *'info'* |
| *TRIM (LEADING | TRAILING | BOTH chars FROM expresie)* | elimină caracterele specificate (*chars*) de la începutul (*leading*) , sfârşitul (*trailing*) sau din ambele părţi, dintr-o expresie caracter dată. | *TRIM (LEADING 'X' FROM*  *'XXXInfoXXX') = 'InfoXXX'*  *TRIM (TRAILING 'X' FROM 'XXXInfoXXX') = 'XXXInfo'*  *TRIM ( BOTH 'X' FROM*  *'XXXInfoXXX') = 'Info'*  *TRIM ( BOTH FROM ' Info*  *') = 'Info'* |
| *LPAD(string, length*  *[,’chars’])* | adaugă *chars* la stânga şirului de caractere *string* până când lungimea noului şir devine *length*; în cazul în care *chars* nu este specificat, atunci se adaugă spaţii libere la stânga lui *string*; | *LPAD (LOWER('iNfO'),6)*  *= ' info'* |
| *RPAD(string, length*  *[,’chars’])* | este similar funcţiei *LPAD*, dar adăugarea de caractere se face la  dreapta şirului; | *RPAD (LOWER('InfO'), 6, 'X') = 'infoXX'* |
| *REPLACE(string1,*  *string2 [,string3])* | întoarce *string1* cu toate apariţiile lui *string2* înlocuite prin *string3*; dacă *string3* nu este specificat, atunci toate apariţiile lui *string2* sunt şterse; | *REPLACE ('$b$bb','$','a') = 'ababb'*  *REPLACE ('$b$bb','$b','ad') =*  *'adadb'*  *REPLACE ('$a$aa','$') = 'aaa'* |
| *UPPER(string), LOWER(string)* | transformă toate literele şirului de caractere *string* în majuscule,  respectiv minuscule; | *LOWER ('InFo') = 'info' UPPER ('iNfO') = 'INFO'* |
| *INITCAP(string)* | transformă primul caracter al şirului în majusculă, restul caracterelor fiind transformate în minuscule | *INITCAP ('iNfO') = 'Info'* |
| *INSTR(string, ‘chars’*  *[,start [,n]])* | caută în *string*, începând de de la poziţia *start*, a *n*-a apariţie a secvenţei *chars* şi întoarce poziţia respectivă; dacă *start* nu este specificat, căutarea se face de la începutul şirului; dacă *n* nu este specificat, se caută prima apariţie a secvenţei *chars*; | *INSTR (LOWER('AbC*  *aBcDe'), 'ab', 5, 2)*  *= 0*  *INSTR (LOWER('AbCdE*  *aBcDe'), 'ab', 5) =*  *7* |
| *ASCII(char)* | furnizează codul *ASCII* al primului caracter al unui şir | *ASCII ('alfa') = ASCII ('a') = 97* |
| *CHR(num)* | întoarce caracterul corespunzător codului *ASCII* specificat | *CHR(97)= 'a'* |
| *CONCAT(string1, string2)* | realizează concatenarea a două şiruri de caractere | *CONCAT ('In', 'fo') = 'Info'* |
| *TRANSLATE(string, source, destination)* | fiecare caracter care apare în şirurile de caractere *string* şi *source* este transformat în caracterul corespunzător (aflat pe aceeaşi poziţie ca şi în *source*) din şirul de caractere *destination* | *TRANSLATE('$a$aa','$','b') = 'babaa'*  *TRANSLATE('$a$aaa','$a','bc')*  *= 'bcbccc'* |

## Funcții pentru prelucrarea datelor calendaristice

|  |  |  |
| --- | --- | --- |
| **Funcţie** | **Descriere** | **Exemplu** |
| *SYSDATE* | întoarce data şi timpul curent | *SELECT SYSDATE*  *FROM dual;*  (de revăzut utilizarea acestei  funcţii împreună cu *TO*\_*CHAR* în cadrul laboratorului 1) |
| *ADD\_MONTHS*  *( expr\_date, nr\_luni)* | întoarce data care este după *nr\_luni* luni de la data  *expr*\_*date*; | *ADD\_MONTHS(’02-APR-*  *2007’, 3)* = ’02-JUL-2007’. |
| *NEXT\_DAY(expr\_date, day)* | întoarce următoarea dată după data *expr*\_*date*, a cărei zi a săptămânii este cea specificată prin şirul de caractere *day* | *NEXT\_DAY(‘18-APR-2007’, ‘Monday’) = ’23-APR-2007’* |
| *LAST\_DAY(expr\_date)* | întoarce data corespunzătoare ultimei zile a lunii din care data *expr*\_*date* face parte | *LAST\_DAY(’02-DEC-2007’) = ’31-DEC-2007’* |
| *MONTHS\_BETWEEN*  *(expr\_date2, expr\_date1)* | întoarce numărul de luni dintre cele două date calendaristice specificate. Data cea mai recentă trebuie specificată în primul argument, altfel rezultatul este negativ. | *MONTHS\_BETWEEN(’02-*  *DEC-2005’, ’10-OCT-2002’) =*  *37.7419355*  *MONTHS\_BETWEEN(’10-*  *OCT-2002’, ’02-DEC-2005’) =*  *-37.7419355* |
| *TRUNC(expr\_date)* | întoarce data *expr*\_*date*, dar cu timpul setat la ora 12:00 AM (miezul nopţii) | *TO\_CHAR(TRUNC*  *(SYSDATE), ‘dd/mm/yy*  *HH24:MI’) = ’02/12/05 00:00’* |
| *ROUND(expr\_date)* | dacă data *expr*\_*date* este înainte de miezul zilei, întoarce data *d* cu timpul setat la ora 12:00 AM; altfel, este returnată data corespunzătoare zilei următoare, cu timpul setat la ora 12:00 AM | TO\_CHAR(ROUND  (SYSDATE*), ‘dd/mm/yy hh24:mi am’) =* ‘*03/12/05*  *00:00 AM’* |
| *LEAST(d1, d2,* …, *dn), GREATEST(d1, d2,* …,  *dn)* | dintr-o listă de date calendaristice, funcţiile întorc prima, respectiv ultima dată în ordine cronologică | *LEAST(SYSDATE, SYSDATE + 3, SYSDATE – 5) = SYSDATE-5*  *GREATEST(SYSDATE,*  *SYSDATE + 3, SYSDATE – 5) = SYSDATE + 3* |

## Operații asupra datelor calendaristice

|  |  |  |
| --- | --- | --- |
| **Operaţie** | **Tipul de date al rezultatului** | **Descriere** |
| *expr*\_*date* -/+ *expr*\_*number* | *Date* | Scade/adună un număr de zile dintr-o / la o dată. Numărul de zile poate sa nu fie întreg (putem adăuga, de exemplu, un număr de minute sau de ore). |
| *expr*\_*date1* – *expr*\_*date2* | *Number* | Intoarce numărul de zile dintre două date calendaristice. Data *expr*\_*date1*  trebuie să fie mai recentă decât *expr*\_*date2*, altfel rezultatul este negativ. |

## Funcții diverse

|  |  |  |
| --- | --- | --- |
| **Funcţie** | **Descriere** | **Exemplu** |
| *DECODE(value, if1, then1, if2, then2, … ,*  *ifN, thenN, else)* | returnează *then1* dacă *value* este egală cu *if1*, *then2* dacă value este egală cu *if2* etc.; dacă *value* nu este egală cu nici una din valorile *if*, atunci funcţia întoarce valoarea *else*; | *DECODE (‘a’, ‘a’, ‘b’, ‘c’) =*  *‘b’*  *DECODE (‘b’, ‘a’, ‘b’, ‘c’) =*  *‘c’*  *DECODE (‘c’, ‘a’, ‘b’, ‘c’) = ‘c’* |
| *NVL(expr\_1, expr\_2)* | dacă *expr\_1* este *NULL,* întoarce *expr\_2*; altfel, întoarce *expr\_1*. Tipurile celor două expresii trebuie să fie compatibile sau *expr\_2* să poată fi convertit implicit la *expr\_1* | *NVL(NULL, 1) = 1*  *NVL(2, 1) = 2*  *NVL('a', 1) = 'a' -- conversie implicită*  *NVL(1, 'a') -- eroare*  *--nu are loc conversia implicită* |
| *NVL2(expr\_1, expr\_2, expr\_3)* | dacă *expr\_1* este *NOT* *NULL*, întoarce *expr\_2*, altfel întoarce *expr\_3* | *NVL2(1, 2, 3) = 2*  *NVL2 (NULL, 1, 2) = 2* |
| *NULLIF (expr\_1, expr\_2)* | Daca *expr\_1* = *expr\_2* atunci funcţia returnează *NULL*, altfel returnează expresia *expr\_1*.  Echivalent cu  *CASE WHEN expr1 = expr2 THEN*  *NULL ELSE expr1 END* | *NULLIF (1, 2) = 1*  *NULLIF (1,1) = NULL* |
| *COALESCE (expr\_1,*  *expr\_2, ... , expr\_n)* | Returnează prima expresie NOT NULL din lista de argumente. | *COALESCE (NULL, NULL,*  *1, 2, NULL) = 1* |
| *UID*, *USER* | întorc *ID*-ul, respectiv *username*-ul utilizatorului *ORACLE* curent | *SELECT USER FROM dual;* |
| *VSIZE(expr)* | întoarce numărul de octeţi ai unei expresii de tip *DATE*, *NUMBER* sau *VARCHAR2* | *SELECT VSIZE(salary)*  *FROM employees*  *WHERE employee\_id=200;* |

## Funcții aritmetice

* unei singure valori, şi aceste funcţii sunt: *ABS* (valoarea absolută), *CEIL* (partea

întreagă superioară), *FLOOR* (partea întreagă inferioară), *ROUND* (rotunjire cu un

număr specificat de zecimale), *TRUNC* (trunchiere cu un număr specificat de

zecimale), *EXP* (ridicarea la putere a lui *e*), *LN* (logaritm natural), *LOG* (logaritm într-o

bază specificată), *MOD* (restul împărţirii a două numere specificate), *POWER*

(ridicarea la putere), *SIGN* (semnul unui număr), *COS* (cosinus), *COSH* (cosinus

hiperbolic), *SIN* (sinus), *SINH* (sinus hiperbolic), *SQRT* (rădăcina pătrată), *TAN*

(tangent), *TANH* (tangent hiperbolic);

* unei liste de valori, iar acestea sunt funcţiile *LEAST* şi *GREATEST*, care întorc cea

mai mică, respectiv cea mai mare valoare a unei liste de expresii.

# JOIN

Pentru *join*, sistemul *Oracle* oferă şi o sintaxă specifică, introdusă de către standardul *SQL3* (*SQL*:*1999*). Această sintaxă nu aduce beneficii în privinţa performanţei faţă de *join*-urile care se specifică în clauza *WHERE*. Tipurile de *join* conforme cu *SQL3* sunt definite prin cuvintele cheie *CROSS JOIN* (pentru produs cartezian), *NATURAL JOIN*, *FULL OUTER JOIN*, clauzele *USING* şi *ON*.

Sintaxa corespunzătoare standardului *SQL3* este următoarea:

***SELECT*** *tabel\_1.nume\_coloană*, *tabel\_2.nume\_coloană*

***FROM*** *tabel\_1*

[***CROSS JOIN*** *tabel\_2*]

*|* [***NATURAL JOIN*** *tabel\_2*]

*|* [***JOIN*** *tabel\_2* ***USING*** (*nume\_coloană*) ]

*|* [***JOIN*** *tabel\_2* ***ON*** (*tabel\_1.nume\_coloană = tabel\_2.nume\_coloană*) ]

*|* [***LEFT*** *|* ***RIGHT*** *|* ***FULL OUTER JOIN*** *tabel\_2*

***ON*** (*tabel\_1.nume\_coloană = tabel\_2.nume\_coloană*) ];

1. *NATURAL JOIN* presupune existenţa unor coloane având acelaşi nume în ambele tabele. Clauza determină selectarea liniilor din cele două tabele, care au valori egale în aceste coloane. Dacă tipurile de date ale coloanelor cu nume identice sunt diferite, va fi returnată o eroare.

Coloanele având acelaşi nume în cele două tabele trebuie să nu fie precedate de numele sau *alias*-ul tabelului corespunzător.

*JOIN tabel\_2 USING nume\_coloană* efectuează un *equijoin* pe baza coloanei cu numele specificat în sintaxă. Această clauză este utilă dacă există coloane având acelaşi nume, dar tipuri de date diferite. Coloanele referite în clauza *USING* ***trebuie să nu conţină calificatori*** (să nu fie precedate de nume de tabele sau *alias*-uri) în nici o apariţie a lor în instrucţiunea *SQL*. Clauzele *NATURAL JOIN* şi *USING* nu pot coexista în aceeaşi instrucţiune *SQL*.

*JOIN tabel\_2 ON tabel\_1.nume\_coloană = tabel\_2.nume\_coloană* efectuează un *equijoin* pe baza condiţiei exprimate în clauza *ON*. Această clauză permite specificarea separată a condiţiilor de *join*, respectiv a celor de căutare sau filtrare (din clauza *WHERE*).

*LEFT*, *RIGHT* şi *FULL OUTER JOIN tabel\_2 ON* (*tabel\_1.nume\_coloană = tabel\_2.nume\_coloană*) efectuează *outer join* la stânga, dreapta, respectiv în ambele părţi pe baza condiţiei exprimate în clauza *ON*.

Un *join* care returnează rezultatele unui *inner join*, dar şi cele ale *outer join*-urilor la stânga şi la dreapta se numeşte *full outer join*.

# Operatorii pe multimi

Operatorii pe mulţimi combină rezultatele obţinute din două sau mai multe interogări. Cererile care conţin operatori pe mulţimi se numesc ***cereri compuse***. Există patru operatori pe mulţimi: *UNION*, *UNION ALL*, *INTERSECT* şi *MINUS*.

Toţi operatorii pe mulţimi au aceeaşi precedenţă. Dacă o instrucţiune *SQL* conţine mai mulţi operatori pe mulţimi, *server*-ul *Oracle* evaluează cererea de la stânga la dreapta (sau de sus în jos). Pentru a schimba această ordine de evaluare, se pot utiliza paranteze.

* ***Operatorul UNION*** returnează toate liniile selectate de două cereri, eliminând duplicatele. Acest operator nu ignoră valorile *null* şi are precedenţă mai mică decât operatorul *IN*.
* Operatorul ***UNION ALL*** returnează toate liniile selectate de două cereri, fără a elimina duplicatele. Precizările făcute asupra operatorului *UNION* sunt valabile şi în cazul operatorului *UNION ALL*. În cererile asupra cărora se aplică *UNION ALL* nu poate fi utilizat cuvântul cheie *DISTINCT*.
* Operatorul ***INTERSECT*** returnează toate liniile comune cererilor asupra cărora se aplică. Acest operator nu ignoră valorile *null*.
* Operatorul ***MINUS*** determină liniile returnate de prima cerere care nu apar în rezultatul celei de-a doua cereri. Pentru ca operatorul *MINUS* să funcţioneze, este necesar ca toate coloanele din clauza *WHERE* să se afle şi în clauza *SELECT*.

# Subcereri

Prin intermediul subcererilor se pot construi interogări complexe pe baza unor instrucţiuni simple.

O subcerere (subinterogare) este o comandă *SELECT* integrată într-o clauză a altei instrucţiuni *SQL*, numită instrucţiune „părinte“ sau instrucţiune exterioară. Subcererile mai sunt numite instrucţiuni *SELECT* imbricate sau interioare.

Rezultatele subcererii sunt utilizate în cadrul cererii exterioare, pentru a determina rezultatul final. În funcţie de modul de evaluare a subcererii în raport cu cererea exterioară, subcererile pot fi:

* nesincronizate (necorelate) sau
* sincronizate (corelate).

Prima clasă de subcereri este evaluată dinspre interior către exterior, adică interogarea externă acţionează pe baza rezultatului cererii interne. Al doilea tip de subcerere este evaluat invers, adică interogarea externă furnizează valori cererii interne, iar rezultatele subcererii sunt transferate cererii externe.

* Subcererile nesincronizate care apar în clauza *WHERE* a unei interogări sunt de forma următoare:

***SELECT*** *expresie1, expresie2, ...* ***FROM*** *nume\_tabel1*

***WHERE*** *expresie\_condiţie operator* (***SELECT*** *expresie*

***FROM*** *nume\_tabel2*);

* + cererea internă este executată prima şi determină o valoare (sau o mulţime de valori);
  + cererea externă se execută o singură dată, utilizând valorile returnate de cererea internă.

* Subcererile sincronizate care apar în clauza *WHERE* a unei interogări au următoarea formă generală:

***SELECT*** *expresie****\_****ext\_1*[, *expresie\_ext\_2 …*]***FROM*** *nume\_tabel\_1 extern*

***WHERE*** *expresie\_condiţie operator*

(***SELECT*** *expresie*

***FROM*** *nume\_tabel\_2*

***WHERE*** *expresie = extern.expresie\_ext*);

* + cererea externă determină o linie candidat;
  + cererea internă este executată utilizând valoarea liniei candidat;
  + valorile rezultate din cererea internă sunt utilizate pentru calificarea sau descalificarea liniei candidat;
  + paşii precedenţi se repetă până când nu mai există linii candidat.

***Obs:*** *operator* poate fi:

* + *single-row operator* (>, =, >=, <, <>, <=), care poate fi utilizat dacă subcererea returnează o singură linie;
  + *multiple-row operator* (*IN*, *ANY*, *ALL*), care poate fi folosit dacă subcererea returnează mai mult de o linie.

Operatorul *NOT* poate fi utilizat în combinaţie cu *IN*, *ANY* şi *ALL*.

Cuvintele cheie *ANY* şi *ALL* pot fi utilizate cu subcererile care produc o singură coloană de valori. Dacă subcererea este precedată de către cuvântul cheie *ALL*, atunci condiţia va fi adevărată numai dacă este satisfăcută de către toate valorile produse de subcerere. Astfel, *<ALL* are semnificaţia „mai mic decât minimul“, iar *>ALL* este echivalent cu „mai mare decât maximul“. Dacă subcererea este precedată de către cuvântul cheie *ANY*, condiţia va fi adevărată dacă este satisfăcută de către oricare (una sau mai multe) dintre valorile produse de subcerere. În comparaţii, *<ANY* are semnificaţia „mai mic decât maximul“; *>ANY* înseamnă „mai mare decât minimul“; *=ANY* este echivalent cu operatorul *IN*.

Dacă subcererea returnează mulţimea vidă, atunci condiţia *ALL* va returna valoarea *true*, iar condiţia *ANY* va returna valoarea *false*. Standardul *ISO* permite utilizarea cuvântului cheie *SOME*, în locul lui *ANY*.

Cuprins

[1. SELECT 1](#_Toc452642725)

[2. Format dată calendaristică 1](#_Toc452642726)

[3. Funcțiile single row 2](#_Toc452642727)

[3.1 Funcții de conversie 2](#_Toc452642728)

[3.2 Funcții pentru prelucrarea caracterelor 2](#_Toc452642729)

[3.3 Funcții pentru prelucrarea datelor calendaristice 4](#_Toc452642730)

[3.4 Operații asupra datelor calendaristice 5](#_Toc452642731)

[3.5 Funcții diverse 5](#_Toc452642732)

[3.6 Funcții aritmetice 6](#_Toc452642733)

[4. JOIN 6](#_Toc452642734)

[5. Operatorii pe multimi 7](#_Toc452642735)

[6. Subcereri 7](#_Toc452642736)