FUNKCIE

Typy funkcií:

- Jednoriadkové funkcie
  - znakové
  - číselné
  - dátumové
  - konverzné
  - funkcie, ktoré prijímajú ako vstup ľuboľný dátový typ
- Skupinové funkcie

Použité výrazy:

stĺpec - názov databázového stĺpca
hodnota - znak, dátum alebo číslo
n - číslo
'reťazec' - znakový reťazec
znaky - niekoľko určitých znakov
dátum - dátumový stĺpec alebo dátumová hodnota

1.1 JEDNORIADKOVÉ FUNKCIE

Spracovávajú každý riadok a pre každý vracajú výsledok. Je ich možné použiť všade tam, kde možno použiť užívateľské premenné, stĺpce, výrazy (napr. za klauzulami SELECT, WHERE, ORDER BY).

1.1.1 Znakové funkcie

Vstupom jednoriadkových znakových funkcií sú znakové dáta. Vracajú ako znakové, tak aj číselné hodnoty.

LOWER (*stlĺpec/hodnota*) spôsobí, že sa všetky písmená *stĺpca alebo hodnoty* prevedú na malé písmená

```
SELECT LOWER('ODDELENIE'), LOWER(dname) FROM dept;
```

```
LOWER(' ODDELENIE')    LOWER(DNAME)
-------------------------------- --------------------
oddelenie              accounting
oddelenie              research
oddelenie              sales
oddelenie              operations
```

UPPER (*stlĺpec/hodnota*) spôsobí, že sa všetky písmená *stĺpca alebo hodnoty* prevedú na veľké písmená

```
SELECT UPPER('oddelenie'), UPPER(dname) FROM dept;
```

```
UPPER(' ODDELENIE')      UPPER(DNAME)
-------------------------------- --------------------
ODDELENIE                ACCOUNTING
ODDELENIE                RESEARCH
ODDELENIE                SALES
ODDELENIE                OPERATIONS
```

INITCAP (*stĺpec/hodnota*) spôsobí, že sa začiatočné písmeno *stĺpca alebo hodnoty* prevedie na veľké písmeno

```
SELECT INITCAP('ODDELENIE'), INITCAP(dname) FROM dept;
```

```
INITCAP(' ODDELENIE')    INITCAP(DNAME)
-------------------------------- ------------------------
Oddelenie                Accounting
Oddelenie                Research
Oddelenie                Sales
Oddelenie                Operations
```

CONCAT (*stĺpec/hodnota, stĺpec/hodnota*) spôsobí zreťazenie *stĺpcov alebo hodnôt* (resp. *stĺpca a hodnoty*), je to alternatíva k operátoru ||

```
SELECT CONCAT('oddelenie',dname) FROM dept;
```

```
CONCAT('ODDELENIE',DNAME)
-----------------------------------------
oddelenieACCOUNTING
oddelenieRESEARCH
oddelenieSALES
oddelenieOPERATIONS
```

LPAD (*stĺpec/hodnota, n, 'reťazec'*) spôsobí doplnenie *stĺpca alebo hodnoty* zľava na dĺžku *n*, pričom sa zľava dopĺňa *'reťazec'*. Ak *'reťazec'* nie je zadaný doplňujú sa medzery.

```
SELECT LPAD('oddelenie',15,'*'), LPAD(deptno, 15,'ABC'), LPAD('oddelenie',15)
FROM dept;
```

```
LPAD('ODDELENIE',15,'*')  LPAD(DEPTNO,15, 'ABC')   LPAD('ODDELENIE',15)
--------------------------------- --------------------------------- ------------------------------
        ******oddelenie          ABCABCABCABCA10          oddelenie
        ******oddelenie          ABCABCABCABCA20          oddelenie
        ******oddelenie          ABCABCABCABCA30          oddelenie
        ******oddelenie          ABCABCABCABCA40          oddelenie
```

RPAD (*stĺpec/hodnota, n, 'reťazec'*) spôsobí doplnenie *stĺpca alebo hodnoty* sprava na dĺžku *n*, pričom sa sprava dopĺňa *'reťazec'*. Ak *'reťazec'* nie je zadaný doplňujú sa medzery.

```
SELECT RPAD('oddelenie',15,'*'), RPAD(deptno, 15,'ABC'), RPAD('oddelenie',15)
FROM dept;
```

```
RRPAD('ODDELENIE',15,'*')  RPAD(DEPTNO,15, 'ABC')   RPAD('ODDELENIE
--------------------------------- --------------------------------- ------------------------------
```

oddelenie******	10ABCABCABCABCA	oddelenie
oddelenie******	20ABCABCABCABCA	oddelenie
oddelenie******	30ABCABCABCABCA	oddelenie
oddelenie******	40ABCABCABCABCA	oddelenie

SUBSTR (*stĺpec/hodnota, pozícia, n*) vráti zo *stĺpca alebo hodnoty* podreťazec začínajúci sa znakom na *pozícii* a dlhý *n* znakov. Ak sa vynechá *n*, vráti sa podreťazec od *pozície* až do konca.

```
SELECT dname, SUBSTR('oddelenie',3,5), SUBSTR(dname,2,7)
FROM dept;
```

DNAME	SUBST('ODDELENIE',3,5)	SUBSTR(DNAME,2,7)
ACCOUNTING	delen	CCOUNTI
RESEARCH	delen	ESEARCH
SALES	delen	ALES
OPERATIONS	delen	PERATIO

INSTR (*stĺpec/hodnota, 'reťazec'*) vráti pozíciu *prvého* výskytu *'reťazca'* v *stĺpci alebo hodnote*
INSTR (*stĺpec/hodnota, 'reťazec', pozícia, n*) vráti pozíciu n-tého výskytu *'reťazca'* v *stĺpci alebo hodnote* začínajúc znakom na *pozícii*

```
SELECT  INSTR('oddelenie','E'), INSTR('oddelenie','DE') , INSTR('oddelenie','E',5,2),
FROM dept;
```

INSTR('ODDELENIE','E')	INSTR('ODDELENIE','DE')	INSTR('ODDELENIE','E',5,2)
4	3	9
4	3	9
4	3	9
4	3	9

LTRIM (*stĺpec/hodnota, znaky*) odstráni zľava vedúce výskyty *znakov* alebo ich kombináciu zo *stĺpca alebo hodnoty* (t.j. *znaky*, resp. ich kombinácia sa odstráni zo začiatku *stĺpca alebo hodnoty*), ak nie sú *znaky* zadané, odstránia sa všetky medzery zľava

```
SELECT dname, LTRIM(dname,'A'), LTRIM(dname,'AS'), LTRIM(dname,'ASOP')
FROM dept;
```

DNAME	LTRIM(DNAME,'A')	LTRIM(DNAME,'AS')	LTRIM(DNAME,'ASOP')
ACCOUNTING	CCOUNTING	CCOUNTING	CCOUNTING
RESEARCH	RESEARCH	RESEARCH	RESEARCH
SALES	SALES	LES	LES
OPERATIONS	OPERATIONS	OPERATIONS	ERATIONS

RTRIM (*stĺpec/hodnota, znaky*) odstráni sprava vedúce výskyty *znakov* alebo ich kombináciu zo *stĺpca alebo hodnoty* (t.j. *znaky*, resp. ich kombinácia sa odstráni z konca *stĺpca alebo hodnoty*), ak nie sú *znaky* zadané, odstránia sa všetky medzery sprava

```
SELECT dname, RTRIM(dname,'G'), RTRIM(dname,'GHS'), RTRIM('oddelenie','sd')
FROM dept;
```

```
DNAME          RTRIM(DNAME,'G')  RTRIM(DNAME,'GHS')  RTRIM('ODDELENIE','SD')
-------------- ----------------- ------------------- -----------------------
ACCOUNTING     ACCOUNTIN         ACCOUNTIN            oddelenie
RESEARCH       RESEARCH          RESEARC              oddelenie
SALES          SALES             SALE                 oddelenie
OPERATIONS     OPERATIONS        OPERATION            oddelenie
```

SOUNDEX (*stĺpec/hodnota*) vráti znakový reťazec predstavujúci fonetickú reprezentáciu *stĺpca alebo hodnoty* (používa sa pre porovnávanie slov, ktoré znejú podobne ale sa inak píšu)

```
SELECT ename, SOUNDEX(ename)
FROM emp
WHERE SOUNDEX(ename)= SOUNDEX('FRED');

ENAME      SOUNDEX(ENAME)
---------- --------------
FORD       F630
```

LENGTH (*stĺpec/hodnota*) vráti počet znakov *stĺpca alebo hodnoty*

```
SELECT dname, LENGTH(dname), LENGTH('oddelenie')
FROM dept;

DNAME          LENGTH(DNAME)  LENGTH('ODDELENIE')
-------------- -------------- -------------------
ACCOUNTING             10                  9
RESEARCH                8                  9
SALES                   5                  9
OPERATIONS             10                  9
```

TRANSLATE (*stĺpec/hodnota, z, do*) zmení v *stĺpci alebo hodnote* na výstupe každý znak *z* na zodpovedajúci (vzhľadom na pozíciu) znak z *do,* pokiaľ nie je zodpovedajúci znak zadaný príslušný znak zo *z* sa odstráni

```
SELECT dname, TRANSLATE(dname,'S', 'X'), TRANSLATE(dname, 'CRS', '12')
FROM dept;

DNAME          TRANSLATE(DNAME,'S','X')  TRANSLATE(DNAME,'CRS','12')
-------------- ------------------------- ----------------------------
ACCOUNTING     ACCOUNTING                A11OUNTING
RESEARCH       REXEARCH                  2EEA21H
SALES          XALEX                     ALE
OPERATIONS     OPERATIONX                OPE2ATION
```

REPLACE (*stĺpec/hodnota, 'reťazec', 'substitučný_reťazec'*) v *stĺpci alebo hodnote* sa na výstupe každý výskyt *'reťazca'* nahradí *'substitučným_reťazcom',* pokiaľ nie je *'substitučný_reťazec'* zadaný odstránia sa všetky výskyty *'reťazca'*.

```
SELECT job, REPLACE(job,'S', 'X'), REPLACE(job,'E')
FROM emp;
```

JOB	REPLACE(JOB,'S','X')	REPLACE(JOB,'E')
CLERK	CLERK	CLRK
SALESMAN	XALEXMAN	SALSMAN
SALESMAN	XALEXMAN	SALSMAN
MANAGER	MANAGER	MANAGR
SALESMAN	XALEXMAN	SALSMAN
MANAGER	MANAGER	MANAGR
MANAGER	MANAGER	MANAGR
ANALYST	ANALYXT	ANALYST
PRESIDENT	PREXIDENT	PRSIDNT
SALESMAN	XALEXMAN	SALSMAN
CLERK	CLERK	CLRK
CLERK	CLERK	CLRK
ANALYST	ANALYXT	ANALYST
CLERK	CLERK	CLRK

14 rows selected.

!!! Jednoriadkové funkcie sa môžu ľubovoľne vnárať, pričom funkcie sa vyhodnocujú od vnútorných smerom k vonkajším funkciám.

1.1.2 Číselné funkcie

Vstupom jednoriadkových číselných funkcií sú číselné hodnoty. Vracajú číselné hodnoty.

ROUND (*stĺpec/hodnota,n*) zaokrúhľuje *stĺpec alebo hodnotu* na *n* desatinných miest. Ak nie je *n* zadané, zaokrúhľuje sa na celé číslo. Ak je *n* záporné, zaokrúhľuje sa naľavo od desatinnej bodky (t.j. na desiatky, stovky, …)

```
SELECT ROUND(sal/32,2) DENNY_PLAT, ROUND(129.921) , ROUND(129.921,-1)
FROM emp;
```

DENNY_PLAT	ROUND(123.921)	ROUND(129.921,-1)
25	124	130
50	124	130
39.06	124	130
92.97	124	130
39.06	124	130
89.06	124	130
76.56	124	130
93.75	124	130
46.88	124	130
34.38	124	130
29.69	124	130
93.75	124	130
40.63	124	130

TRUNC (*stĺpec/hodnota,n*) orezáva *stĺpec alebo hodnotu* na *n* desatinných miest. Ak nie je *n* zadané, oreže sa na celé číslo. Ak je *n* záporné, číslice naľavo od desatinnej bodky sa zmenia na nulu

```
SELECT TRUNC(sal/32,2) DENNY_PLAT, TRUNC(123.921,1), TRUNC(123.921),
       TRUNC(129.921,-1)
FROM emp;
```

DENNY_PLAT	TRUNC(123.921,1)	TRUNC(123.921)	TRUNC(129.921,-1)
25	123.9	123	120
50	123.9	123	120
39.06	123.9	123	120
92.96	123.9	123	120
39.06	123.9	123	120
89.06	123.9	123	120
76.56	123.9	123	120
93.75	123.9	123	120
156.25	123.9	123	120
46.87	123.9	123	120
34.37	123.9	123	120
29.68	123.9	123	120
93.75	123.9	123	120
40.62	123.9	123	120

14 rows selected.

CEIL (*stĺpec/hodnota*) nájde najmenšie celé číslo väčšie alebo rovné *stĺpcu alebo hodnote* (t.j. hornú celú časť)

```
SELECT sal/32, CEIL(sal/32), CEIL(123.921), CEIL(123.1), CEIL(-129.921)
FROM emp;
```

SAL/32	CEIL(SAL/32)	CEIL(123.921)	CEIL(123.1)	CEIL(-129.921)
25	25	124	124	-129
50	50	124	124	-129
39.0625	40	124	124	-129
92.96875	93	124	124	-129
39.0625	40	124	124	-129
89.0625	90	124	124	-129
76.5625	77	124	124	-129
93.75	94	124	124	-129
156.25	157	124	124	-129
46.875	47	124	124	-129
34.375	35	124	124	-129
29.6875	30	124	124	-129
93.75	94	124	124	-129
40.625	41	124	124	-129

14 rows selected.

FLOOR (*stĺpec/hodnota*) nájde najväšie celé číslo menšie alebo rovné *stĺpcu alebo hodnote* (t.j. dolná celá časť)

```
SELECT sal/32, FLOOR(sal/32), FLOOR(123.921), FLOOR(123.1), FLOOR(-129.921)
FROM emp;
```

SAL/32	FLOOR(SAL/32)	FLOOR(123.921)	FLOOR(123.1)	FLOOR(-129.921)
25	25	123	123	-130
50	50	123	123	-130
39.0625	39	123	123	-130
92.96875	92	123	123	-130
39.0625	39	123	123	-130
89.0625	89	123	123	-130
76.5625	76	123	123	-130
93.75	93	123	123	-130
156.25	156	123	123	-130
46.875	46	123	123	-130
34.375	34	123	123	-130
29.6875	29	123	123	-130
93.75	93	123	123	-130
40.625	40	123	123	-130

POWER (*stĺpec/hodnota,n*) umocní *stĺpec alebo hodnotu* na *n*-tú mocninu. (*n* môže byť aj záporné číslo, ale musí byť vždy celé)

```
SELECT sal, POWER(sal,2), POWER(2,3), POWER(4,-0.5), POWER(123,0)
FROM emp;
```

SAL	POWER(SAL,2)	POWER(2,3)	POWER(4,-0.5)	POWER(123,0)
800	640000	8	.5	1
1600	2560000	8	.5	1
1250	1562500	8	.5	1
2975	8850625	8	.5	1
1250	1562500	8	.5	1
2850	8122500	8	.5	1
2450	6002500	8	.5	1
3000	9000000	8	.5	1
5000	25000000	8	.5	1
1500	2250000	8	.5	1
1100	1210000	8	.5	1
950	902500	8	.5	1
3000	9000000	8	.5	1
1300	1690000	8	.5	1

14 rows selected.

EXP (*n*) umocní e na *n*-tú mocninu. (e=2.71828183)

```
SELECT EXP(4)   FROM dual;

    EXP(4)
-------------
 54.59815
```

SQRT (*stĺpec/hodnota*) nájde druhú odmocninu *stĺpca alebo hodnoty*.

```
SELECT sal, SQRT(sal), SQRT(4)   FROM emp;

SAL         SQRT(SAL)       SQRT(4)
---------   ---------------  ---------
  800          28.284271          2
 1600          40                 2
 1250          35.355339          2
 2975          54.543561          2
 1250          35.355339          2
 2850          53.385391          2
 2450          49.497475          2
 3000          54.772256          2
 5000          70.710678          2
 1500          38.729833          2
 1100          33.166248          2
  950          30.82207           2
 3000          54.772256          2
 1300          36.055513          2

14 rows selected.
```

SIGN (*stĺpec/hodnota*) vráti -1, ak je *stĺpec alebo hodnota* záporné číslo; vráti 1, ak je *stĺpec alebo hodnota* kladné číslo

ABS (*stĺpec/hodnota*) vráti absolútnu hodnotu *stĺpca alebo hodnoty*

MOD (*stĺpec1/hodnota1, stĺpec2/hodnota2*) vráti zvyšok po delení *stĺpcov alebo hodnôt* (resp. *stĺpca a hodnoty*)

```
SELECT sal, comm, MOD(sal,comm), MOD(sal,32), MOD(50,40)
FROM emp
WHERE deptno=30;

SAL    COMM   MOD(SAL,COMM) MOD(SAL,32)    MOD(50,40)
--------- ---------  ---------------------- -------------------  -----------------
 1250      500          250                  2                  10
 1250     1400         1250                  2                  10
 1500        0         1500                 28                  10
  950                                       22                  10
```

Je možné použiť aj ïalšie matematické funkcie:

LOG (*m,n*) vráti logaritmus čísla *n* o základe *m*
SIN (*n*) vráti sínus čísla *n*
SINH (*n*) vráti hyperbolický sínus čísla *n*
TAN (*n*) vráti tangens čísla *n*
TANH (*n*) vráti hyperbolický tangens čísla *n*
COS (*n*) vráti kosínus čísla *n*
COSH (*n*) vráti hyperbolický kosínus čísla *n*

1.1.3 Dátumové funkcie

Dátumové funkcie pracujú s dátami ORACLE. Vracajú hodnoty dátumového typu DATE, okrem funkcie MONTHS_BETWEEN, ktorá vracia číslo. Oracle ukladá dátum vo svojom vnútornom číselnom formáte. Ten reprezentuje storočie, rok, mesiac, deň, hodiny, minúty a sekundy. Dátum v oracle môže byť z intervalu od 1. januára 4712 pred naším letopočtom až do 31. 12. 4711 nášho letopočtu.
Implicitný formát dátumu v ORACLE je DD-MON-YY, kde DD je číslo dňa, YY je číslo roka (posledné dvojčíslie) a MON je textová skratka mesiaca.

SYSDATE je funkcia, ktorá vracia aktuálny dátum a čas. Obvykle sa vyberá SYSDATE z tabuľky dual, ktorá má len jeden stĺpec DUMMY a jeden riadok 'X'. Preto sa používa v prípadoch, ke chceme získať nejakú hodnotu iba raz.

```
SELECT SYSDATE FROM dual;

SYSDATE
---------
10-OCT-97
```

Pri dátume možno používať nasledovné operácie:

dátum + číslo	pripočíta k *dátumu* počet dní určených *číslom*
dátum - číslo	odčíta od *dátumu* počet dní určených *číslom*
dátum - dátum	odčíta jeden *dátum* od druhého
dátum + číslo/24	pripočíta k *dátumu* počet hodín určených ako *číslo/24*

Výsledkom rozdielu dátumov je číslo predstavujúce počet dní medzi danými dátumami. Výsledkom ostatných operácií je dátum.

```
SELECT hiredate, hiredate+7, hiredate-3, SYSDATE-hiredate
FROM emp
WHERE deptno=30;

HIREDATE  HIREDATE+7  HIREDATE-7  SYSDATE-HIREDATE
--------- ----------- ----------- ----------------
22-FEB-81 01-MAR-81   19-FEB-81    5727.6893
28-SEP-81 05-OCT-81   25-SEP-81    5509.6893
08-SEP-81 15-SEP-81   05-SEP-81    5529.6893
03-DEC-81 10-DEC-81   30-NOV-81    5443.6893
```

MONTHS_BETWEEN (*dátum1, dátum2*) zistí počet mesiacov medzi *dátumom1 a dátumom2*. Výsledok môže byť kladný (ak je *dátum1* neskorší ako *dátum2*) alebo záporný (ak je *dátum1* skorší ako *dátum2*).

```
SELECT ename, MONTHS_BETWEEN(SYSDATE, hiredate)
FROM emp
ORDER BY MONTHS_BETWEEN(SYSDATE, hiredate);
```

ENAME	MONTHS_BETWEEN(SYSDATE,HIREDATE)
ADAMS	165.56051
SCOTT	166.65729
MILLER	177.20567
JAMES	178.85083
FORD	178.85083
KING	179.39922
MARTIN	181.04438
TURNER	181.68954
CLARK	184.65729
BLAKE	185.91535
JONES	186.88309
WARD	188.23793
ALLEN	188.30245
SMITH	190.39922

14 rows selected.

ADD_MONTHS (*dátum, n*) pripočíta *n* mesiacov k *dátumu*, *n* musí byť celé číslo a môže byť kladné aj záporné.

```
SELECT ADD_MONTHS(SYSDATE,-5), ADD_MONTHS(hiredate,3)
FROM emp
WHERE deptno=30;
```

ADD_MONTH(SYSDATE,-5)	ADD_MONTH(hiredate,3)
29-MAY-96	22-MAY-81
29-MAY-96	28-DEC-81
29-MAY-96	08-DEC-81
29-MAY-96	03-MAR-82

NEXT_DAY (*dátum, znaky*) vráti dátum najbližšieho dňa v týždni určeného *znakmi,* ktorý nasleduje po *dátume.* Znakmi môže byť číslo označujúce deň alebo znakový reťazec označujúci názov dňa v týždni

```
SELECT hiredate, NEXT_DAY(hiredate, 'FRIDAY'), NEXT_DAY(hiredate, 6)
FROM emp
WHERE deptno=30;
```

HIREDATE	NEXT_DAY(HIREDATE, 'FRIDAY')	NEXT_DAY(HIREDATE, 6)

22-FEB-81	27-FEB-81	27-FEB-81
28-SEP-81	02-OCT-81	02-OCT-81
08-SEP-81	11-SEP-81	11-SEP-81
03-DEC-81	04-DEC-81	04-DEC-81

LAST_DAY (*dátum*) vráti dátum posledného dňa v mesiaci, ktorý obsahuje *dátum*

```
SELECT hiredate, LAST_DAY(hiredate), LAST_DAY('01-JUN-72')
FROM emp
WHERE deptno=30;
```

HIREDATE	NEXT_DAY(HIREDATE, 'FRIDAY')	NEXT_DAY(HIREDATE, 6)
22-FEB-81	28-FEB-81	30-JUN-72
28-SEP-81	30-SEP-81	30-JUN-72
08-SEP-81	30-SEP-81	30-JUN-72
03-DEC-81	31-DEC-81	30-JUN-72

ROUND(*dátum, 'reťazec'*)

ak reťazec='MONTH', vráti prvý deň mesiaca obsahujúceho *dátum,* ak je *dátum* v prvej polovici mesiaca, inak vracia prvý deň nasledujúceho mesiaca,

ak reťazec='YEAR', vráti prvý deň roku obsahujúceho *dátum,* ak je *dátum* v prvej polovici roku, inak vracia prvý deň nasledujúceho roku,

ak je reťazec nie je zadaný, vráti *dátum* s časom nastaveným na 12.00 AM (polnoc), čo je výhodné pre porovnávanie dátumov s rôznymi časmi,

```
SELECT sysdate, ROUND(sysdate, 'MONTH'), ROUND(sysdate, 'YEAR')
FROM dual;
```

SYSDATE	ROUND(SYSDATE)	ROUND(SYSDATE)
05-NOV-96	01-NOV-96	01-JAN-97

TRUNC(*dátum, 'reťazec'*)

ak reťazec='MONTH', vráti dátum prvého dňa mesiaca obsahujúceho *dátum,*

ak reťazec='YEAR', vráti prvý deň roku obsahujúceho *dátum,*

```
SELECT sysdate, TRUNC(sysdate, 'MONTH'), TRUNC(sysdate, 'YEAR')
FROM dual;
```

SYSDATE	ROUND(SYSDATE)	ROUND(SYSDATE)
05-NOV-96	01-NOV-96	01-JAN-96

1.1.4 Konverzné funkcie

Slúžia na prevody dátových typov (hodnotu jedného dátového typu na druhý dátový typ).

TO_NUMBER(*znaky*) prevádza *znaky* obsahujúce číslo na číslo (typ NUMBER)

```
SELECT ename, job, sal
FROM emp
WHERE sal > TO_NUMBER('2000');
```

```
ENAME      JOB             SAL
----------    ---------------   ---------
JONES      MANAGER      2975
BLAKE      MANAGER      2850
CLARK      MANAGER      2450
SCOTT      ANALYST       3000
KING       PRESIDENT    5000
FORD       ANALYST       3000

6 rows selected.
```

TO_CHAR(*číslo/dátum, ['format']*) prevádza *číslo alebo dátum* na znakový formát *'formát',* ak nie je zadaný *'formát'*, tak *číslo* ostane v takom tvare ako je a *dátum* ostáva v implicitnom tvare (DD-MON-YY). (Popisy formátov sa nachádzajú neskôr v texte).

```
SELECT ename, job, sal
FROM emp
WHERE TO_CHAR(sal)>'4500';
```

```
ENAME   JOB             SAL
----------   ---------         ---------
SMITH    CLERK          800
KING     PRESIDENT   5000
JAMES    CLERK          950
```

(t.j. vyberú sa iba tí zamestnanci, ktorích prvá číslica platu je väčšie ako 4, druhá číslica platu je väčšie ako 5 atï. Ak *číslo* má viac číslic ako číslo v porovnávacom reťazci ('4500'), tak sa *číslo* považuje za väčšie)

```
SELECT TO_CHAR(sysdate, 'Day, DD.MM.YY') FROM dual;
```

```
TO_CHAR(SYSDATE,'DAY,DD.MM.YY')
-----------------------------------------------------------
Wednesday, 06.11.96
```

(t.j. systémový dátum sa prevedie z implicitného tvaru (DD-MON-YY) na tvar *'format'* (Day, DD.MM.YY))

- *'format'* musí byť vždy uzavretý v apostrofoch,

- ak sa vypisuje celý názov mesiaca alebo názov dňa v týždni, hodnoty na výstupe sú doplnené medzerami na dĺžku 9 (najdlhšie názvy dní a mesiacov v angličtine), na odstránenie výplne medzerami použite predponu FM (File Mode - Vyplňovací režim)
- FM možno použiť aj na potlačenie vedúcich núl vo formáte ddth (t.j. 05th sa zmení na 5th)
- dátum bude zobrazený takými písmenami ako *'format':*

SELECT TO_CHAR(sysdate, 'Day'), TO_CHAR(sysdate, 'DAY'), TO_CHAR(sysdate, 'day') FROM dual

```
TO_CHAR(SYSDATE,'DAY')          TO_CHAR(SYSDATE,'DAY')
TO_CHAR(SYSDATE,'DAY')
------------------------------------ ------------------------------------ ----------
----------
Wednesday                WEDNESDAY           wednesday
```

- ak chceme zistiť časovú zložku dňa a zobraziť vo *'formate':*

```
SELECT TO_CHAR(sysdate, 'HH:MI:SS') FROM dual

TO_CHAR(SYSDATE,'HH:MI:SS')
-------------------------------------------
07:35:49
```

- ak chceme vypísať číslo v určitom *'formate':*

```
SELECT TO_CHAR(sal, '$9,999') FROM emp WHERE deptno=30

TO_CHAR
---------------
 $1,250
 $1,250
 $1,500
   $950
```

!!! *'formát'* ovplyvňuje iba výpis (spôsob zobrazenia), ale nie vnútornú reprezentáciu hodnoty stĺpca

TO_DATE(*znaky, 'format'*) prevádza *znaky* predstavujúce dátum na dátumový formát '*formát',* ak nie je zadaný *'formát'*, tak dátum bude v implicitnom tvare (DD-MON-YY)

```
SELECT ename, hiredate
FROM emp
WHERE hiredate=TO_DATE('December 17,1980', 'Month DD,YYYY');

ENAME      HIREDATE
-----------  ---------------
SMITH       17-DEC-80
```

- táto funkcia sa používa aj pre vkladanie dátumu do databázy v inom ako implicitnom formáte:

```
INSERT INTO emp (empno, deptno, hiredate)
```

VALUES (7777, 20, TO_DATE('17.12.1980', 'DD.MM.YYYY');

Číselné formáty

Obraz	Význam		Príklad
9	číselná pozícia (počet deviatok určuje šírku zobrazenia)	999999	1234
0	zobrazenie "vedúcich núl"	099999	001234
$	pohyblivý znak dolára	$999999	$1234
.	desatinná bodka na danom mieste	99999.99	12345.10
,	oddeľovač rádu tisícok	999,999	1,234
MI	znamienko mínus vpravo (záporné hodnoty)	99999MI	1234-
PR	záporné čísla zobrazené v hranatých zátvorkách	99999PR	<1234>
EEEE	vedecká notácia (formát musí obsahovať 4 znaky E)	99,999EEEE	1,234E+03
V	násobiť 10^n (n=počet deviatok za V)	9999V99	123400
B	zobrazovať nulové hodnoty ako medzery, nie ako nuly	B9999,99	1234,00

- tieto číselné formáty sa dajú používať aj v príkaze COLUMN

Dátumové formáty

Obraz	Význam
SCC | SC	storočie, 'S' znamená, že sa pred dátum pr. n. l. vloží '-'
YYYY | SYYYY	rok, 'S' znamená, že sa pred dátum pr. n. l. vloží '-'
YYY | YY | Y	posledné 3, 2 alebo 1 číslice roku
Y,YYY	rok s čiarkou na príslušnej pozícii
SYEAR | YEAR	rok hláskovane, 'S' znamená, že sa pred dátum pr. n. l. vloží '-'
BC | AD	indikátor pr. n. l./n. l.
Q	štvrťrok
MM	mesiac
MONTH	názov mesiaca, doplnený medzerami na dĺžku 9 znakov
MON	trojpísmenová skratka názvu mesica
WW | W	týždeň roku či mesiaca
DDD | DD | D	deň roku, mesiaca alebo týždňa

DAY	názov dňa, vyplnený medzerami na dĺžku 9 znakov
DY	trojpísmenová skratka názvu dňa
J	Juliánsky dátum, počet dní od 31.decembra 4713 pr. n. l.
AM | PM	indikátor poludnia (dopoludnie, odpoludnie)
A.M. | P.M.	indikátor poludnia s bodkami
HH | HH12	hodina dňa (1-12)
HH24	hodina dňa (0-23)
MI	minúta
SS	sekunda
SSSSS	sekundy po polnoci (0-86399)
/.,etc.	znamienka sa prenesú do výsledku
"…."	Reťazec v úvodzovkách sa prenesie do výsledku

Predpona, ktorú možno použiť ku dátumovým formátom:

fm 'File Mode' - Vyplňovací režim. Potláča vyplňovanie medzerami pre formát MONTH alebo YEAR, resp. vedúce nuly vo formáte ddth
Ďalší výskyt *fm* opäť nastaví vyplňovanie medzerami

Prípony, ktoré možno použiť k dátumovým formátom:

TH Poradové číslo (napr. DD*th* vypíše 4th)

SP Hláskové číslo (napr. DD*SP* vypíše FOUR)

SPTH Hláskové poradové číslo (napr. DD*SPTH* vypíše FOURTH)

!!! Na výpis (zobrazenie) majú vplyv veľké alebo malé písmená:

DAY MONDAY	MONTH JULY	ddTH (DDTH)	14TH
Day Monday	Month July	ddTh (DDTh)	14Th
day Monday	month july	ddth (DDth)	14th

1.1.5 Funkcie, ktoré akceptujú ako vstup ľubovoľný dátový typ

DECODE (*stĺpec/výraz, zdroj1, náhrada1, [zdroj2, náhrada2, …], implicitná hodnota*) uľahčuje podmienené dotazy (robí to isté ako príkazy typu 'CASE' alebo 'IF-THEN-ELSE'). *Stĺpec/výraz* sa porovná s každou hodnotou *zdroja* a je vrátená *náhrada*, ak *stĺpec/výraz* sa rovná hodnote *zdroja.* Ak sa *stĺpec/výraz* sa nerovná hodnote *zdroja,* vráti sa *implicitná hodnota.* Ak nie je *implicitná hodnota* zadaná, vráti sa hodnota NULL.

```
SELECT grade, DECODE(grade, '1','20%',    '2','15%',   '3','10%',   '5%') odmena
FROM salgrade;
```

```
GRADE   ODMENA
--------   ----------
     1     20%
     2     15%
     3     10%
     4      5%
     5      5%
```

!!! vrátená hodnota bude prevedená na rovnaký dátový typ, ako má *náhrada*

NVL(*stĺpec, hodnota*) konvertuje hodnotu NULL *stĺpca* na danú *hodnotu. Stĺpec aj hodnota* musia byť rovnakého dátového typu.

```
SELECT NVL(comm,0) FROM emp WHERE deptno=30;

NVL(COMM,0)
-----------
        500
       1400
          0
          0
```

GREATEST(*stĺpec1/hodnota1, stĺpec2/hodnota2*) vracia väčšiu z hodnôt na zozname. Všetky hodnoty *stĺpca2/hodnoty2* sa pred porovnávaním konvertujú na typ *stĺpca1/hodnoty1*

```
SELECT GREATEST(comm, sal), GREATEST(1500, 150)
FROM emp
WHERE deptno=30;

GREATEST(COMM,SAL) GREATEST(1500,150)
------------------ ------------------
              1250               1500
              1400               1500
              1500               1500
                                 1500
```

LEAST(*stĺpec1/hodnota1, stĺpec2/hodnota2*) vracia menšiu z hodnôt na zozname. Všetky hodnoty *stĺpca2/hodnoty2* sa pred porovnávaním konvertujú na typ *stĺpca1/hodnoty1*

```
SELECT LEAST(comm, sal), LEAST(1500, 150)
FROM emp
WHERE deptno=30;

LEAST(COMM,SAL) LEAST(1500,150)
--------------- ---------------
            500             150
           1250             150
              0             150
                            150
```

VSIZE(*stĺpec/hodnota*) vracia počet bytov *stĺpca/hodnoty* v internej reprezentácii ORACLE

```
SELECT VSIZE(sal), VSIZE(1500), VSIZE(ename) FROM emp WHERE deptno=30;

VSIZE(SAL) VSIZE(1500) VSIZE(ENAME)
---------- ----------- ------------
         3           2            4
         3           2            6
         2           2            6
```

3 2 5

1.2 SKUPINOVÉ (AGREGOVANÉ) FUNKCIE

Pomocou týchto funkcií môžeme získať súhrnné informácie pre skupiny riadkov.

Funkcia	Vrátená hodnota
AVG([DISTINCT/ALL] *n*)	priemerná hodnota *n*, nedefinované hodnoty sa ignorujú
COUNT([DISTINCT/ALL] výraz/*)	číslo určujúce, koľkokrát dáva *výraz*hodnotu inú než NULL * spôsobí, že sa spočítajú všetky vybrané riadky, vrátane duplicitných riadkov a riadkov s hodnotou NULL
MAX([DISTINCT/ALL] *výraz*)	maximálna hodnota *výrazu*
MIN([DISTINCT/ALL] *výraz*)	minimálna hodnota *výrazu*
STDDEV([DISTINCT/ALL] *n*)	štandardná odchýlka *n*, hodnoty NULL sa ignorujú
SUM([DISTINCT/ALL] *n*)	súčet hodnôt *n*, hodnoty NULL sa ignorujú
VARIANCE([DISTINCT/ALL] *n*)	variancia hodnôt, nedefinovanéhodnoty sa ignorujú

- skupinové funkcie pracujú s viacerými riadkami
- DISTINCT spôsobí, že sa v skupinovej funkcii berú do úvahy iba rôzne (neduplicitné) hodnoty. ALL (implicitná hodnota) berie do úvahy všetky hodnoty vrátane duplicitných
- tam, kde je uvedený *výraz*, môžu mať argumenty dátový typ CHAR, NUMBER alebo DATE
- všetky skupinové funkcie okrem COUNT(*) ignorujú hodnoty NULL, ak sa majú brať do úvahy aj hodnoty NULL, treba použiť funkciu NVL

!!! Pre skupinové funkcie slúžia klauzulky GROUP BY a HAVING príkazu SELECT