

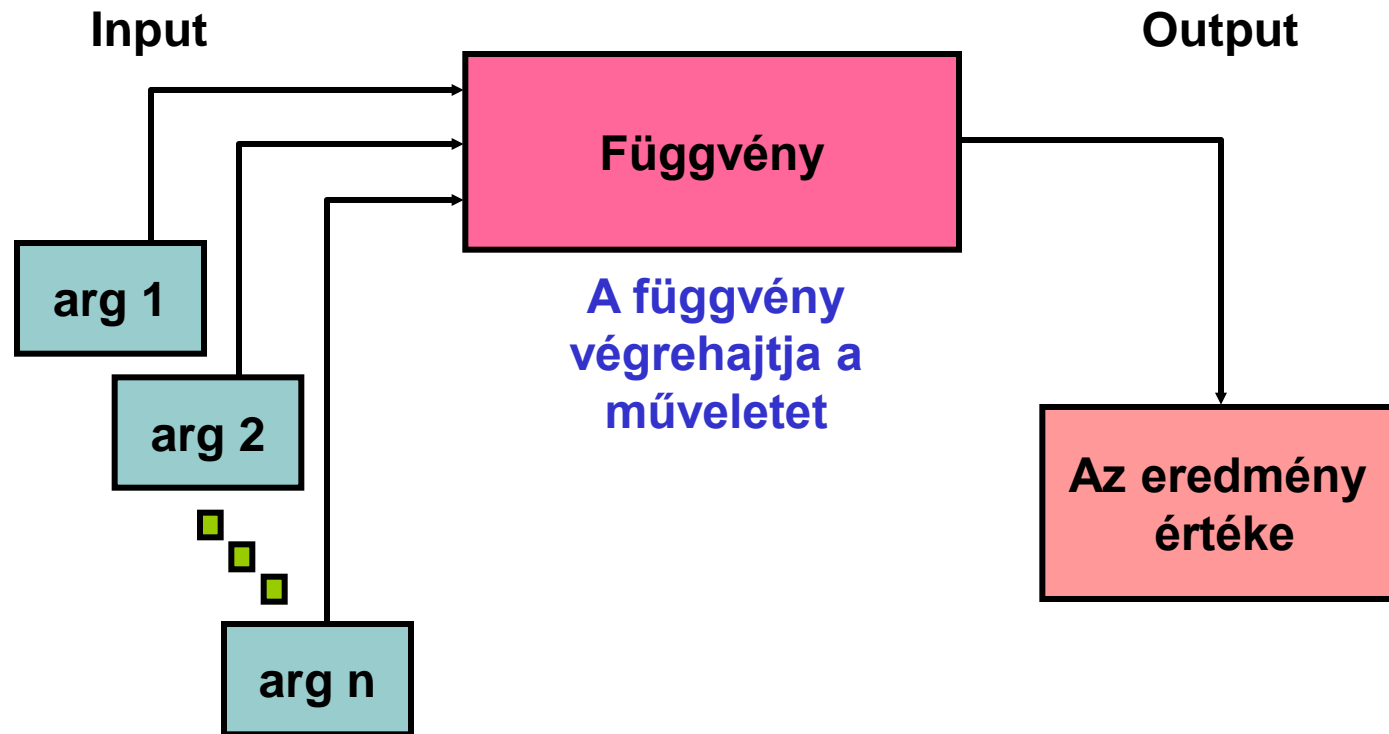
**Leckék Oracle gyakorlatra  
./sql/lecke03\_fuggvenyek.pdf**

SQL sorfüggvények

# Célkitűzés

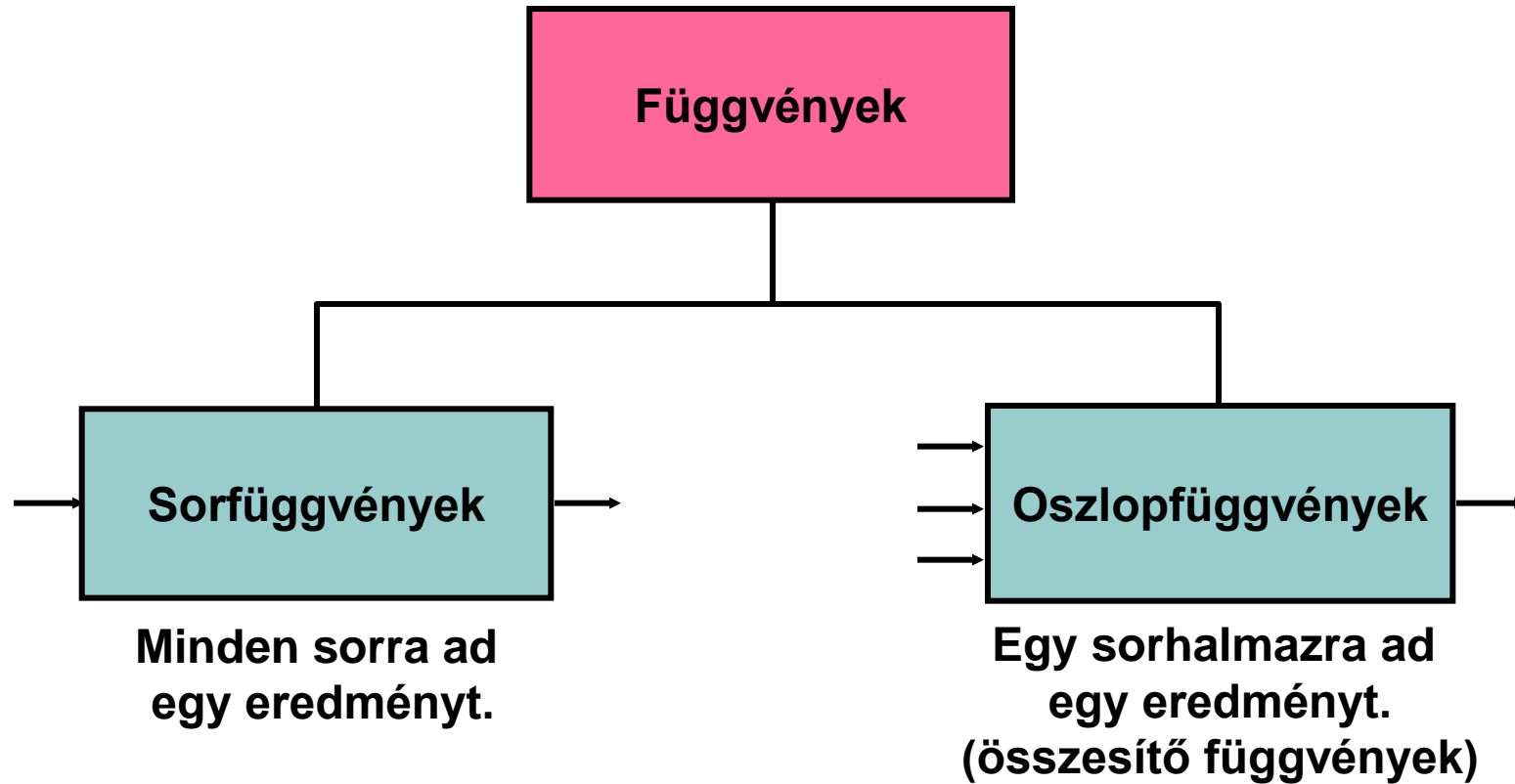
- Különböző típusú SQL sorfüggvények megismerése
- A karaktertípusú, numerikus, illetve dátumtípusú sorfüggvények használata a `SELECT` utasításokban
- Típus-átalakító sorfüggvények megismerése

# SQL-függvények



**A bemutatott függvények többsége Oracle-specifikus.**

# Az SQL-függvények két típusa

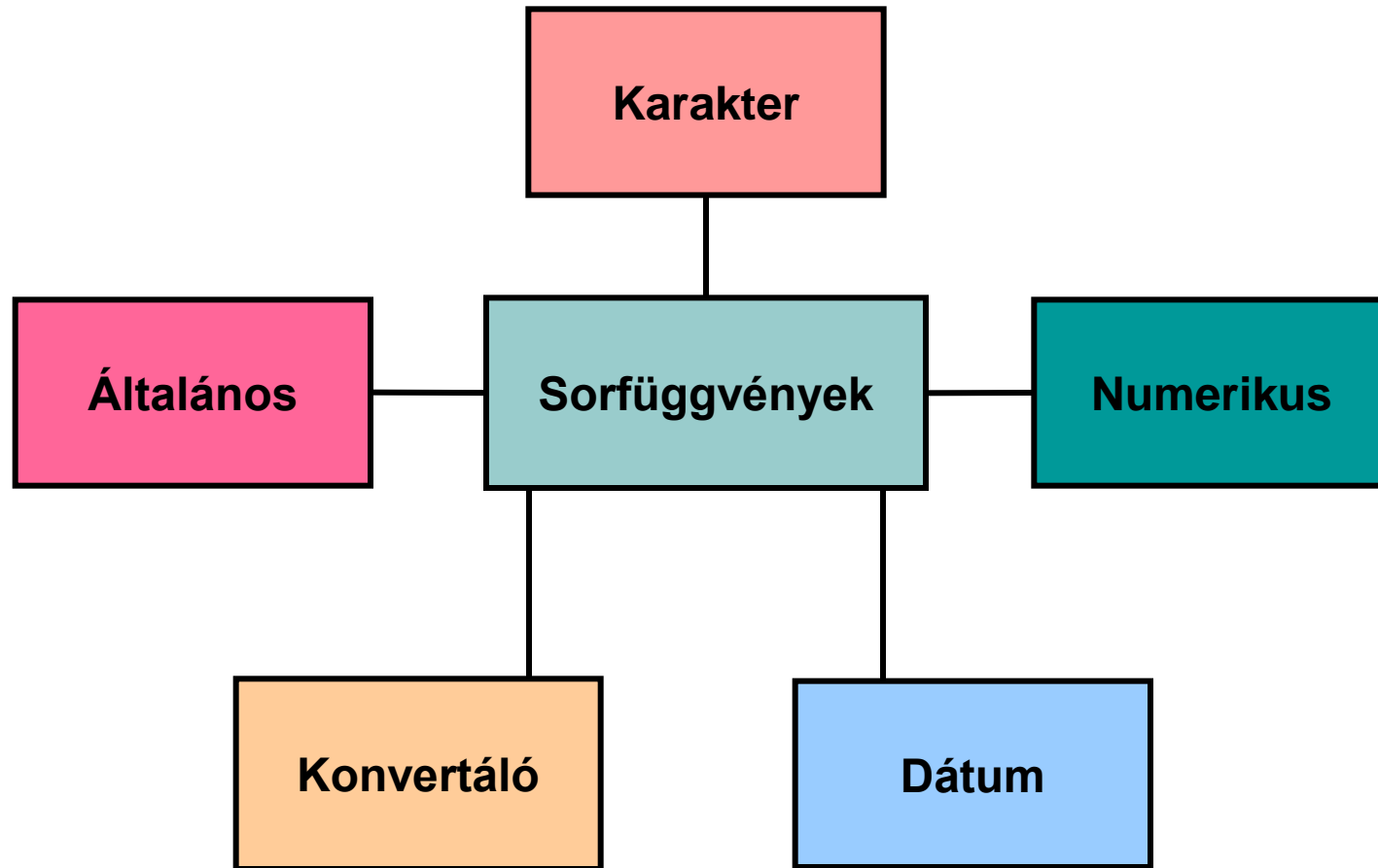


# Sorfüggvények

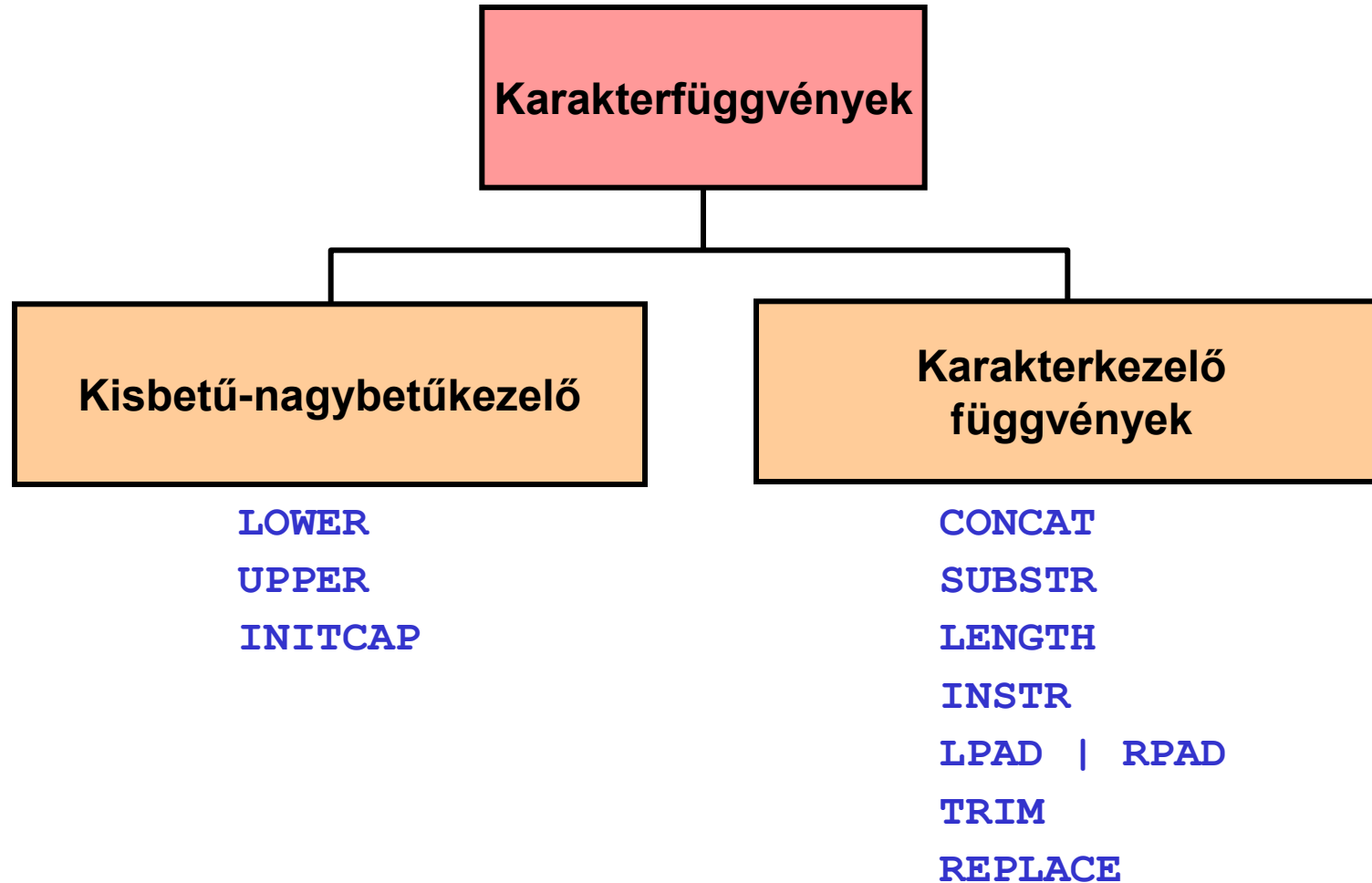
- A sorfüggvények:
  - Az adattételek átalakítására, feldolgozására használhatók.
  - Több argumentumból egy értéket eredményez.
  - Az argumentum lehet felhasználói konstans, változó, oszlopnév, kifejezés.
  - A lekérdezés eredményének minden sorára meghívódik.
  - Minden sorra egy értéket ad vissza.
  - Az eredménye más adattípusú is lehet mint az argumentum.
  - Egymásba lehet ágyazni.
  - Használható a SELECT, WHERE és ORDER BY részekben.

```
function_name [(arg1, arg2, ...)]
```

# Sorfüggvények



# Karakterfüggvények



# Kisbetű-nagybetű kezelő függvények

- Például:

Függvény	Eredmény
LOWER('SQL Nyelv')	sql nyelv
UPPER('SQL Nyelv')	SQL NYELV
INITCAP('SQL Nyelv')	Sql Nyelv

```
SELECT 'The job id for' || UPPER(last_name) || ' is '  
      || LOWER(job_id) AS "EMPLOYEE DETAILS"  
FROM   employees;
```

**EREDMÉNYE:**

EMPLOYEE DETAILS
The job id for KING is ad_pres
The job id for KOCHHAR is ad_vp
The job id for DE HAAN is ad_vp



# A kisbetű-nagybetű kezelő függvények használata

- Adjuk meg Higgins azonosítóját, nevét és osztályának azonosítóját:

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE last_name = 'higgins';
no rows selected
```

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE LOWER(last_name) = 'higgins';
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
205	Higgins	110

Ugyanezt adná a következő is:

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE INITCAP(last_name) = 'Higgins';
```

# Karakterkezelő függvények használata

Function	Result
<code>CONCAT('Hello', 'World')</code>	HelloWorld
<code>SUBSTR('HelloWorld',1,5)</code>	Hello
<code>LENGTH('HelloWorld')</code>	10
<code>INSTR('HelloWorld', 'W')</code>	6
<code>LPAD(salary,10, '*')</code>	*****24000
<code>RPAD(salary, 10, '*')</code>	24000*****
<code>REPLACE('JACK and JUE', 'J', 'BL')</code>	BLACK and BLUE
<code>TRIM('H' FROM 'HelloWorld')</code>	elloWorld

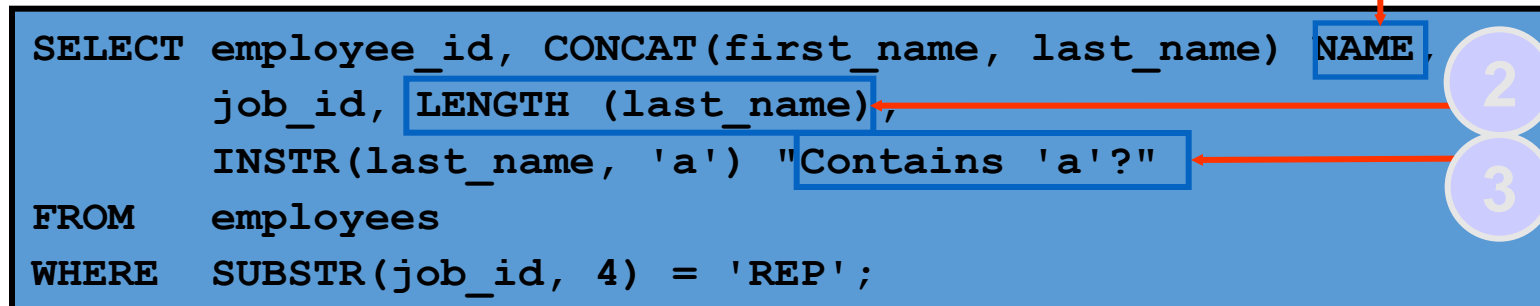
**A függvényekbe helyettesítő változókat is tehetünk:**

```
select upper('&valami') from dual
```

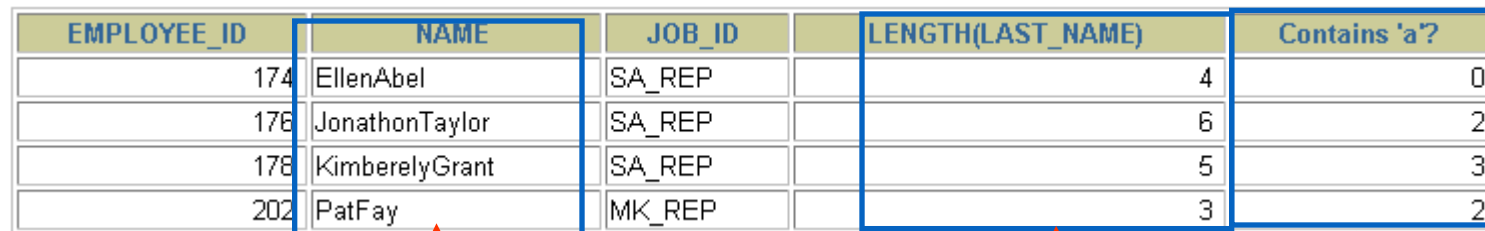
# Karakterkezelő függvények használata

1. **Vonjuk össze a keresztnévet és vezetéknévet!**
2. **Hány betűs a vezetéknév?**
3. **A vezetéknévben hanyadik betű "a"?**
4. **A beosztáskód a 4. betűtől 'REP'.**

```
SELECT employee_id, CONCAT(first_name, last_name) NAME,  
       job_id, LENGTH(last_name),  
       INSTR(last_name, 'a') "Contains 'a'?"  
FROM employees  
WHERE SUBSTR(job_id, 4) = 'REP';
```



EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
174	EllenAbel	SA_REP	4	0
176	JonathonTaylor	SA_REP	6	2
178	KimberelyGrant	SA_REP	5	3
202	PatFay	MK_REP	3	2



```
SELECT * FROM EMPLOYEES  
WHERE SUBSTR(last_name, -1, 1) = 'n'; -- a vezetéknév n-re végződik.
```

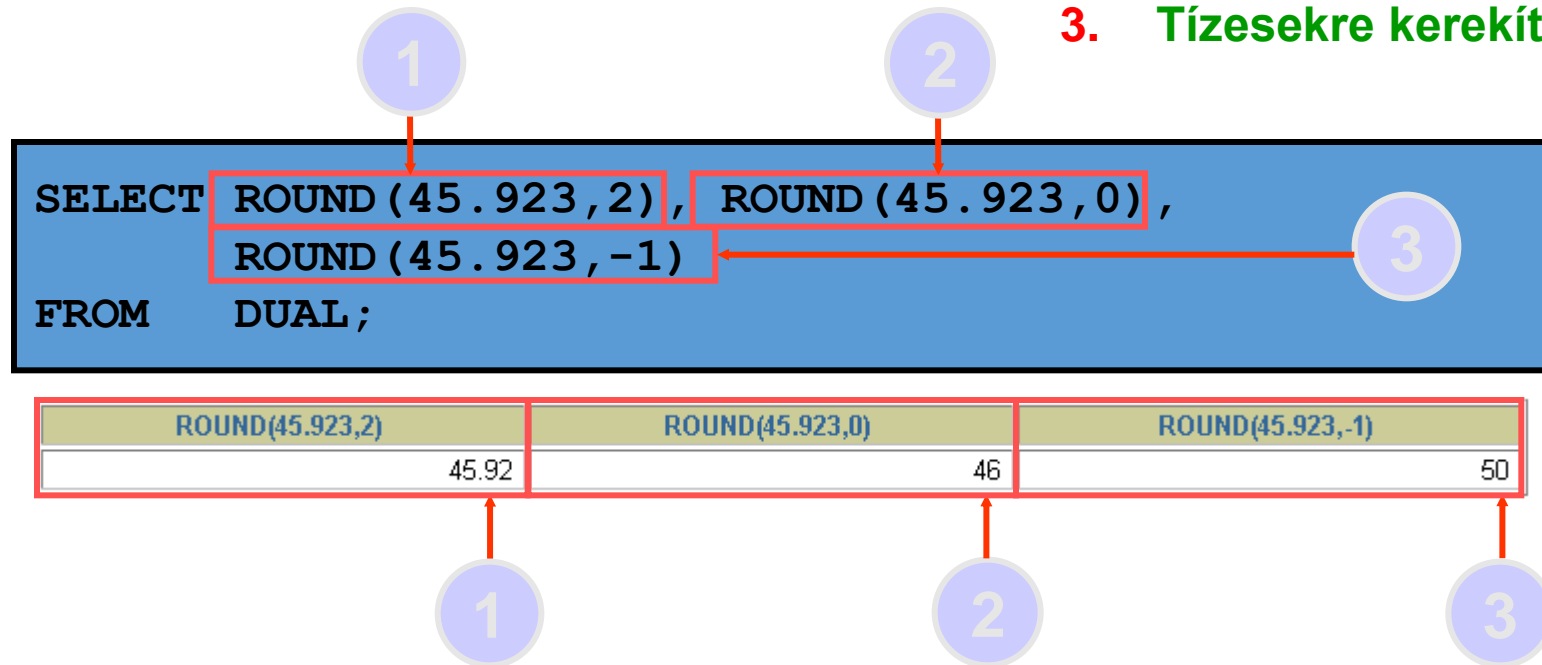
# Numerikus függvények

- **ROUND:** Adott tizedes jegyre kerekít (ha n negatív, akkor a tizedesvesszőtől balra kerekít).
- **TRUNC:** Adott tizedesjegy utáni részt levágja
- **MOD:** A maradékos osztást maradékát adja vissza

Függvény	Eredmény
ROUND (45.926, 2)	45.93
TRUNC (45.926, 2)	45.92
MOD (1600, 300)	100

# A ROUND függvény használata

1. Két tizedesjegyre kerekítünk
2. Egészekre kerekítünk
3. Tízessekre kerekítünk.

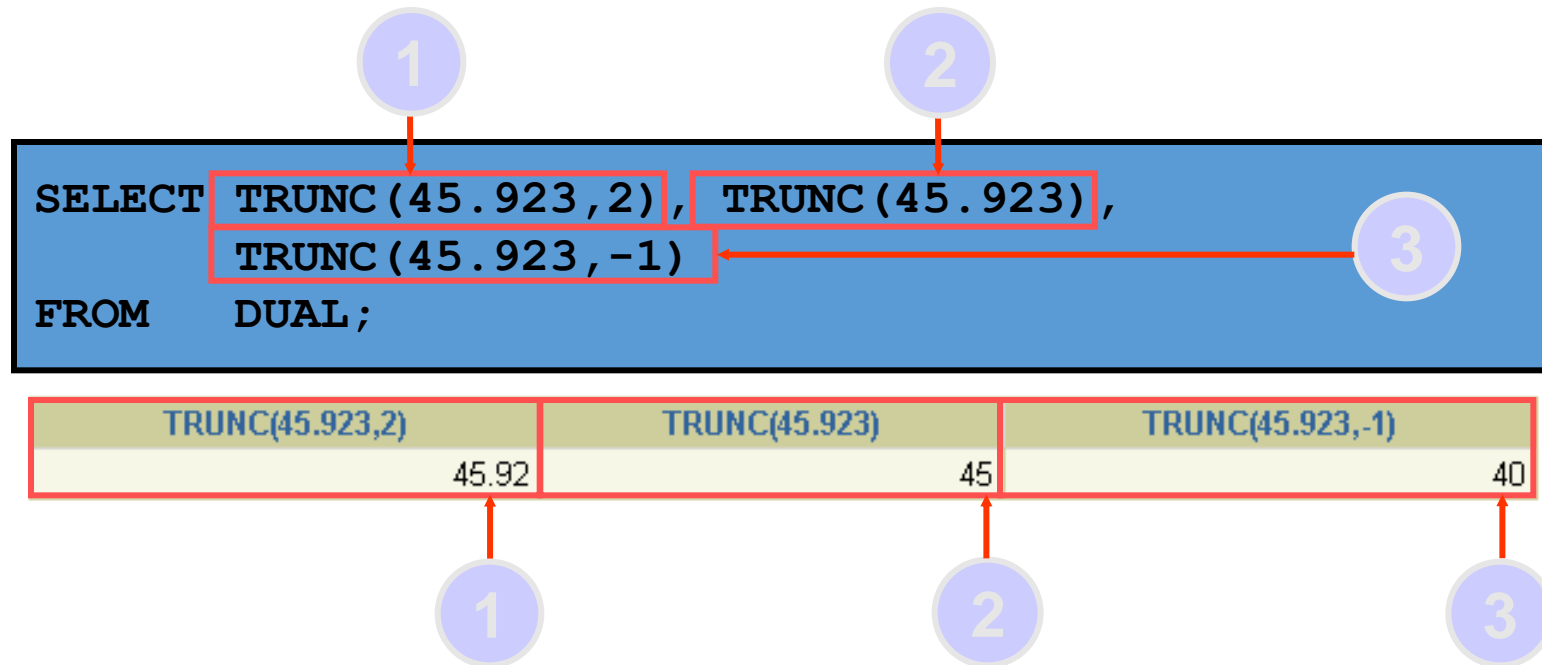


A **DUAL** tábla a **SYS** tulajdona, nyilvánossá van téve, így tesztelésre jól használható.

Egy **DUMMY** nevű oszlopa van, egy sora, amiben **X** szerepel.

# A TRUNC függvény használata

1. **Két tizedesjegyre csonkolunk**
2. **Egészekre csonkolunk (elhagyjuk a törtrészt).**
3. **Tízesekre csonkolunk.**



A ROUND és TRUNC dátumokra is használható.

# A MOD függvény használata

- Mennyi 5000-rel osztva a fizetések maradéka a kereskedőkre, azaz az 'SA\_REP' beosztású dolgozókra?

```
SELECT last_name, salary, MOD(salary, 5000)
FROM employees
WHERE job_id = 'SA_REP';
```

LAST_NAME	SALARY	MOD(SALARY,5000)
Abel	11000	1000
Taylor	8600	3600
Grant	7000	2000

Gyakran használjuk egy egész szám paritásának eldöntésére.

# A dátumok használata

- A dátumokat az Oracle numerikusan tárolja. A dátum tartalmazza az évszázadot, évet, hónapot, napot, órát és másodpercet.
- A dátum megjelenítésének alapértelmezése: **DD-MON-YY**.
  - Ha az aktuális dátum a század második felében van, és a kétjegyű évszám az első felében, akkor a következő századnak tekinti, különben az aktuális századnak.
  - Ha az aktuális dátum a század első felében van, és a kétjegyű évszám a második felében, akkor az előző századnak tekinti, különben az aktuális századnak.

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date < '01-FEB-88';
```

LAST_NAME	HIRE_DATE
King	17-JUN-87
Whalen	17-SEP-87



# A dátumok használata

- A `SYSDATE` függvény segítségével megkaphatjuk:
  - az adatbázis-kezelő rendszerdátumát és
  - az adatbázis-kezelő rendszeridejét.

Ha a hónapok nevét magyarul akarjuk látni:

```
ALTER SESSION set NLS_LANGUAGE = "HUNGARIAN"
```

```
select to_char(sysdate,'yyyy-Month-dd') from dual
```

```
TO_CHAR(SYSDATE,'YYYY-MONTH-DD')
```

```
2008-Február -16
```

# Dátumaritmetika

- Egy dátumhoz hozzá lehet adni vagy ki lehet vonni egy számot. A számnak megfelelő nappal növeli vagy csökkenti a dátum értékét.
- Két dátum kivonása a köztük eltelt napok számát adja vissza.
- Mivel egy óra a nap 24-ed része, így órákat is hozzá tudunk adni egy dátumhoz.

# Dátumműveletek használata

Adjuk meg a 90-es osztályon, hogy hány hetet dolgoztak a belépés óta a dolgozók!

```
SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS  
FROM employees  
WHERE department_id = 90;
```

LAST_NAME	WEEKS
King	744.245395
Kochhar	626.102538
De Haan	453.245395

# Dátumfüggvények

Függvény	Eredmény
<code>MONTHS_BETWEEN (date1, date2)</code>	A dátumok közti hónapok száma
<code>ADD_MONTHS (date, n)</code>	n hónappal növeli a dátumot
<code>NEXT_DAY (date, 'char')</code>	A következő adott nevű nap dátuma.
<code>LAST_DAY (date)</code>	A dátum hónapjának utolsó napja.
<code>ROUND (date [, 'fmt'] )</code>	A dátum kerekítése
<code>TRUNC (date [, 'fmt'] )</code>	A dátum levágása

# A dátumfüggvények használata

Függvény	Eredmény
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-FEB-95')	'28-FEB-95'

```
SELECT employee_id, hire_date,  
       MONTHS_BETWEEN (SYSDATE, hire_date) TENURE,  
       ADD_MONTHS (hire_date, 6) REVIEW,  
       NEXT_DAY (hire_date, 'FRIDAY'),  
       LAST_DAY(hire_date)  
FROM employees  
WHERE MONTHS_BETWEEN (SYSDATE, hire_date) < 70;
```

EMPLOYEE_ID	HIRE_DATE	TENURE	REVIEW	NEXT_DAY(	LAST_DAY(
107	07-FEB-99	31.6982407	07-AUG-99	12-FEB-99	28-FEB-99
124	16-NOV-99	22.4079182	16-MAY-00	19-NOV-99	30-NOV-99
149	29-JAN-00	19.9885633	29-JUL-00	04-FEB-00	31-JAN-00
178	24-MAY-99	28.1498536	24-NOV-99	28-MAY-99	31-MAY-99

# Dátumfüggvények használata

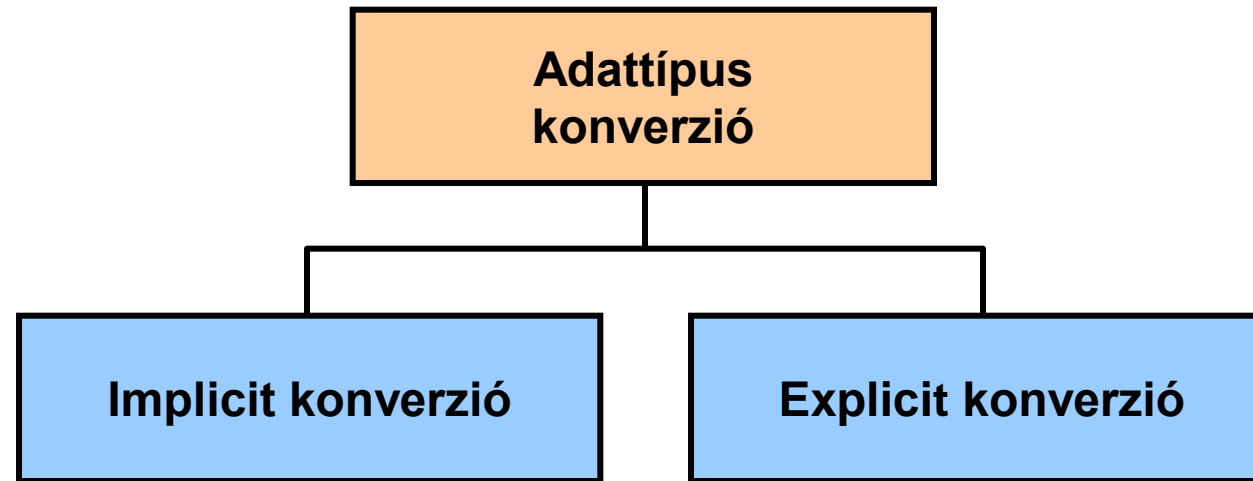
Tegyük fel, hogy `SYSDATE = '25-JUL-03'`:

Függvény	Eredmény
<code>ROUND (SYSDATE , 'MONTH' )</code>	<code>01-AUG-03</code>
<code>ROUND (SYSDATE , 'YEAR' )</code>	<code>01-JAN-04</code>
<code>TRUNC (SYSDATE , 'MONTH' )</code>	<code>01-JUL-03</code>
<code>TRUNC (SYSDATE , 'YEAR' )</code>	<code>01-JAN-03</code>

```
SELECT employee_id, hire_date,  
       ROUND(hire_date, 'MONTH'), TRUNC(hire_date, 'MONTH')  
FROM employees  
WHERE hire_date LIKE '%97';
```

EMPLOYEE_ID	HIRE_DATE	ROUND(HIR	TRUNC(HIR
142	29-JAN-97	01-FEB-97	01-JAN-97
202	17-AUG-97	01-SEP-97	01-AUG-97

# Konvertáló függvények



A hasonló adattípusok konverzióját az Oracle szerverre is bízhatjuk (**implicit**), de ajánlott inkább konvertáló függvényeket használni (**explicit**).

# Implicit adattípus-konverzió

- A következő **típusok** konverzióját az Oracle szerver automatikusan elvégzi:

Miről	Mire
VARCHAR2 vagy CHAR	NUMBER
VARCHAR2 vagy CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

```
SELECT hire_date FROM hr.employees WHERE hire_date > '1990-01-01';
```

A jobb oldal karakteres, a bal oldal dátum, mégis érvényes az összehasonlítás.

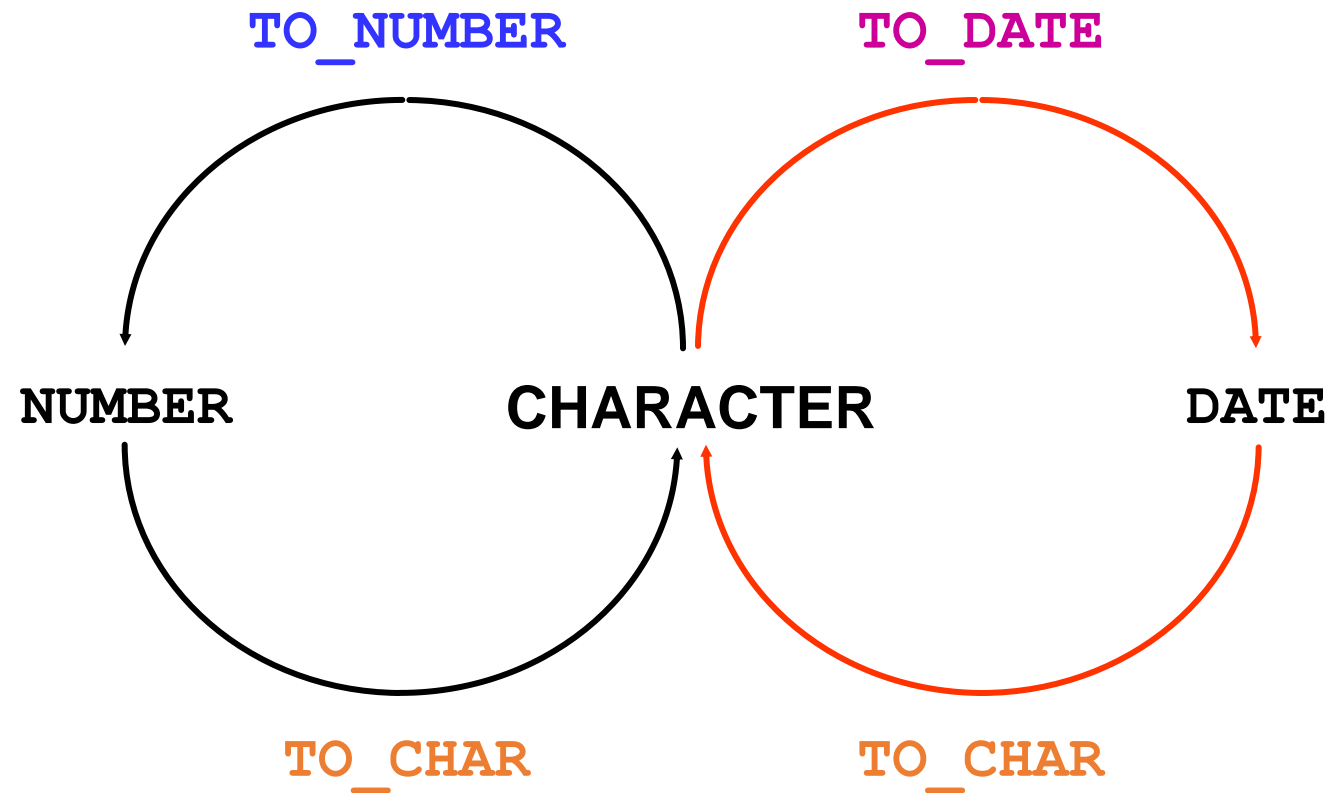


# Implicit adattípus-konverzió

- A következő típusú **kifejezések** konverzióját az Oracle szerver automatikusan elvégzi:

<b>Miről</b>	<b>Mire</b>
VARCHAR2 vagy CHAR	NUMBER
VARCHAR2 vagy CHAR	DATE

# Explicit adattípus-konverzió



# Explicit adattípus-konverzió

<b>Függvény</b>	<b>Leírás</b>
<b>TO_CHAR(number date,[ fmt], [nlsparams])</b>	<b>A VARCHAR2 karakter formátumát az <i>fmt</i> modellel lehet megadni. Az nlsparams paraméter mondja meg, hogy milyen tízedesvesszőt, ezres csoportosítót, pénznemeket használunk.</b>
<b>TO_CHAR(number date,[ fmt], [nlsparams])</b>	<b>Dátumkonverzió esetén az nlsparams paraméter mondja meg, hogy milyen nyelven adtuk meg a napok, hónapok nevét, vagy miként rövidítettük a neveket.</b>
<b>TO_NUMBER(char,[fmt], [nlsparams])</b>	<b>Az fmt és nlsparams opcionális paraméterek értelme a fentiek szerint.</b>
<b>TO_DATE(char,[fmt],[nlsparams])</b>	<b>Az fmt és nlsparams opcionális paraméterek értelme a fentiek szerint.</b>

# A TO\_CHAR függvény használata dátummal

```
TO_CHAR(date, 'format_model')
```

- A formátum megadása:
  - egyszeres idézőjelek között
  - kisbetű érzékeny
  - tetszőleges érvényes dátumformátumot tartalmazhat
  - Az `fm` elemmel lehet az automatikusan kiegészített szóközöket eltávolítani, illetve a bevezető nullákat elnyomni

```
SELECT employee_id, TO_CHAR(hire_date, 'MM/YY')  
Month_Hired  
FROM employees  
WHERE last_name = 'Higgins';
```

EMPLOYEE_ID	MONTH
205	06/94

# A dátumformátum leggyakoribb elemei

Elem	Értelme
YYYY	Évszám (számokkal)
YEAR	Évszám (szövegesen)
MM	Hónap sorszáma
MONTH	Hónap teljes neve
MON	Hónap 3 betűvel rövidítve
DY	A hét napja 3 betűvel rövidítve
DAY	A hét napjának teljes neve
DD	A nap sorszáma a hónapban

**select to\_char(sysdate,'Year') from dual**

**TO\_CHAR(SYSDATE,'YEAR')**

**Two Thousand Eight**

# További formátum modellek

<b>Elem</b>	<b>Leírás</b>
<b>SCC or CC</b>	<b>Century; server prefixes B.C. date with -</b>
<b>Years in dates YYYY or SYYYY</b>	<b>Year; server prefixes B.C. date with -</b>
<b>YYY or YY or Y</b>	<b>Last three, two, or one digits of year</b>
<b>Y,YYY</b>	<b>Year with comma in this position</b>
<b>IYYY, IYY, IY, I</b>	<b>Four-, three-, two-, or one-digit year based on the ISO standard</b>
<b>SYEAR or YEAR</b>	<b>Year spelled out; server prefixes B.C. date with -</b>
<b>BC or AD</b>	<b>Indicates B.C. or A.D. year</b>
<b>B.C. or A.D.</b>	<b>Indicates B.C. or A.D. year using periods</b>
<b>Q</b>	<b>Quarter of year</b>
<b>MM</b>	<b>Month: two-digit value</b>
<b>MONTH</b>	<b>Name of month padded with blanks to length of nine characters</b>
<b>MON</b>	<b>Name of month, three-letter abbreviation</b>
<b>RM</b>	<b>Roman numeral month</b>
<b>WW or W</b>	<b>Week of year or month</b>
<b>DDD or DD or D</b>	<b>Day of year, month, or week</b>
<b>DAY</b>	<b>Name of day padded with blanks to a length of nine characters</b>
<b>DY</b>	<b>Name of day; three-letter abbreviation</b>
<b>J</b>	<b>Julian day; the number of days since December 31, 4713 B.C.</b>

# A dátum típusú formátum modell használata

- Időformátum megadása:

HH24:MI:SS AM	15:45:32 PM
---------------	-------------

Szöveget kettős idézőjelek között lehet a dátumban használni:

DD "of" MONTH	12 of OCTOBER
---------------	---------------

- A számokat szövegesen is kiírathatjuk:

ddspth	fourteenth
--------	------------

# További formátummodellek

<b>Element</b>	<b>Description</b>
AM or PM	Meridian indicator
A.M. or P.M.	Meridian indicator with periods
HH or HH12 or HH24	Hour of day, or hour (1–12), or hour (0–23)
MI	Minute (0–59)
SS	Second (0–59)
SSSSS	Seconds past midnight (0–86399)

<b>Element</b>	<b>Description</b>
/ . ,	Punctuation is reproduced in the result.
“of the”	Quoted string is reproduced in the result.

<b>Element</b>	<b>Description</b>
TH	Ordinal number (for example, DDTH for 4TH)
SP	Spelled-out number (for example, DDSF for FOUR)
SPTH or THSP	Spelled-out ordinal numbers (for example, DDSPTH for FOURTH)



# A TO\_CHAR függvény használata dátumokkal

```
SELECT last_name,  
       TO_CHAR(hire_date, 'fmDD Month YYYY')  
       AS HIREDATE  
FROM   employees;
```

LAST_NAME	HIREDATE
King	17 June 1987
Kochhar	21 September 1989
De Haan	13 January 1993
Hunold	3 January 1990
Ernst	21 May 1991
Lorentz	7 February 1999
Mourgos	16 November 1999

```
SELECT last_name,  
       TO_CHAR(hire_date, 'fmDdspth "of" Month YYYY fmHH:MI:SS AM')  
       HIREDATE FROM employees;
```

LAST_NAME	HIREDATE
King	Seventeenth of June 1987 12:00:00 AM
Kochhar	Twenty-First of September 1989 12:00:00 AM

# A TO\_CHAR függvény használata számokkal

```
TO_CHAR(number, 'format_model')
```

- A legfontosabb formátummodellek:

Elem	Eredménye
9	Szám
0	Nulla
\$	Lebegő dollárjel
L	Lebegő pénznem
.	Tízedespont
,	Ezresek elválasztójele

## További formátummodellek

<b>Element</b>	<b>Description</b>	<b>Example</b>	<b>Result</b>
9	Numeric position (number of 9s determine display width)	999999	1234
0	Display leading zeros	099999	001234
\$	Floating dollar sign	\$999999	\$1234
L	Floating local currency symbol	L999999	FF1234
D	Returns in the specified position the decimal character. The default is a period (.).	99D99	99.99
.	Decimal point in position specified	999999.99	1234.00
G	Returns the group separator in the specified position. You can specify multiple group separators in a number format model.	9,999	9G999
,	Comma in position specified	999,999	1,234
MI	Minus signs to right (negative values)	999999MI	1234-
PR	Parenthesize negative numbers	999999PR	<1234>
EEEE	Scientific notation (format must specify four Es)	99.999EEEE	1.234E+03
U	Returns in the specified position the "Euro" (or other) dual currency	U9999	€1234
V	Multiply by 10 <i>n</i> times ( <i>n</i> = number of 9s after V)	9999V99	123400
S	Returns the negative or positive value	S9999	-1234 or +1234
B	Display zero values as blank, not 0	B9999.99	1234.00

# A TO\_CHAR függvény használata számokkal

```
SELECT TO_CHAR(salary, '$99,999.00') SALARY  
FROM employees  
WHERE last_name = 'Ernst';
```

SALARY
\$6,000.00

Az Oracle szerver (#) jeleket tesz, ha több számjegy van mint amennyit megadtunk a formátum modellben.

# A TO\_NUMBER és a TO\_DATE függvények használata

- Karakterként megadott számokat lehet a TO\_NUMBER függvénnyel számtípusá alakítani:

```
TO_NUMBER(char[, 'format_model'])
```

- A szöveggként megadott dátumot a TO\_DATE függvénnyel lehet dátumtípusá konvertálni:

```
TO_DATE(char[, 'format_model'])
```

- Ezekben a függvényekben használhatjuk az `fx` módosítót. Ennek jelentése, hogy pontosan meg kell egyezni méretre is (szóközöket is figyelembe véve) az argumentumoknak.

# RR dátumformátum

Aktuális év	Megadott dátum	RR forma	YY forma
1995	27-OCT-95	1995	1995
1995	27-OCT-17	2017	1917
2001	27-OCT-17	2017	2017
2001	27-OCT-95	1995	2095

		Ha a megadott kétjegyű év:	
		0–49	50–99
Ha az aktuális év két utolsó jegye	0–49	Aktuális évszázad dátuma	Az aktuális előtti évszázad dátuma
	50–99	Az aktuális utáni évszázad dátuma	Az aktuális évszázad dátuma

# Az RR dátumformátum használata

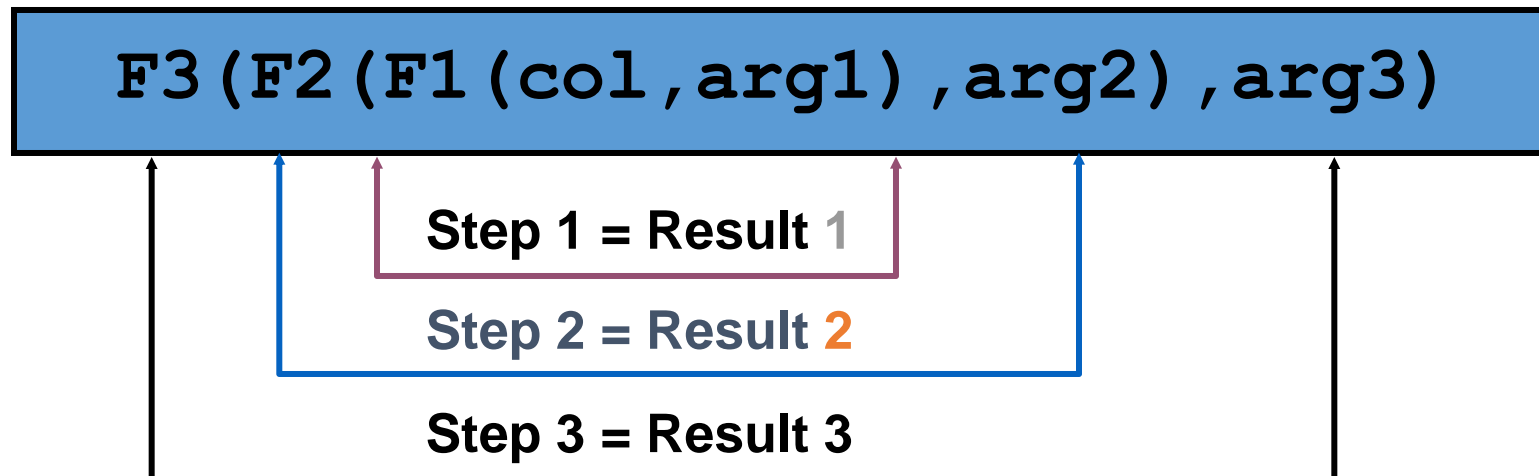
- Keressük meg az 1990 előtt belépett dolgozókat! Ha RR dátumformátumot használunk, akkor mindegy hogy 1999-ben adjuk ki az utasítást vagy 2008-ban:

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-YYYY')  
FROM employees  
WHERE hire_date < TO_DATE('01-Jan-90', 'DD-Mon-RR');
```

LAST_NAME	TO_CHAR(HIR
King	17-Jun-1987
Kochhar	21-Sep-1989
Whalen	17-Sep-1987

# Függvények egymásba ágyazása

- A sorfüggvények tetszőleges mélységig egymásba ágyazhatók.
- A kiértékelés belülről kifelé történik.





# Függvények egymásba ágyazása

```
SELECT last name,  
       UPPER(CONCAT(SUBSTR (LAST_NAME, 1, 8), '_US'))  
FROM   employees  
WHERE  department_id = 60;
```

LAST_NAME	UPPER(CONCAT(SUBSTR(LAST_NAME,1,8
Hunold	HUNOLD_US
Ernst	ERNST_US
Lorentz	LORENTZ_US

# Általános függvények

- Ezek a függvények tetszőleges adattípussal és nullértékek esetén is működnek.
  - NVL (expr1, expr2)
  - NVL2 (expr1, expr2, expr3)
  - NULLIF (expr1, expr2)
  - COALESCE (expr1, expr2, ..., exprn)

# Az NVL függvény

- A nullértéket a megadott értékkel helyettesíti:
  - Az adattípus lehet dátum, karakter, szám.
  - Az argumentumok adattípusának egyezőnek kell lenniük:
    - `NVL(commission_pct, 0)`
    - `NVL(hire_date, '01-JAN-97')`
    - `NVL(job_id, 'No Job Yet')`

# Az NVL függvény használata

```
SELECT last name, salary, NVL(commission_pct, 0),  
       (salary*12) + (salary*12*NVL(commission_pct, 0)) AN_SAL  
FROM employees;
```

LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
King	24000	0	288000
Kochhar	17000	0	204000
De Haan	17000	0	204000
Hunold	9000	0	108000
Ernst	6000	0	72000
Lorentz	4200	0	50400
Mourgos	5800	0	69600
Rajs	3500	0	42000

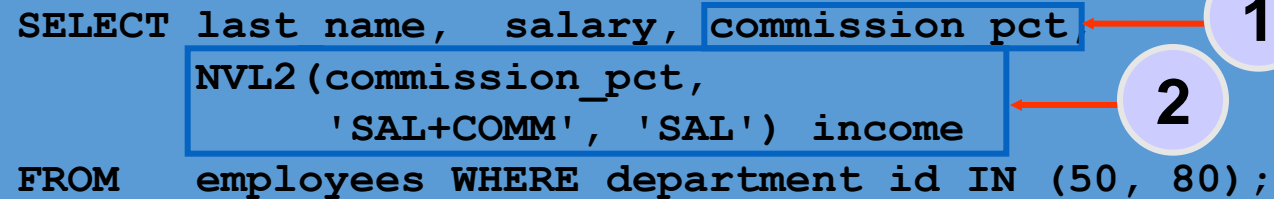
...  
20 rows selected.

1

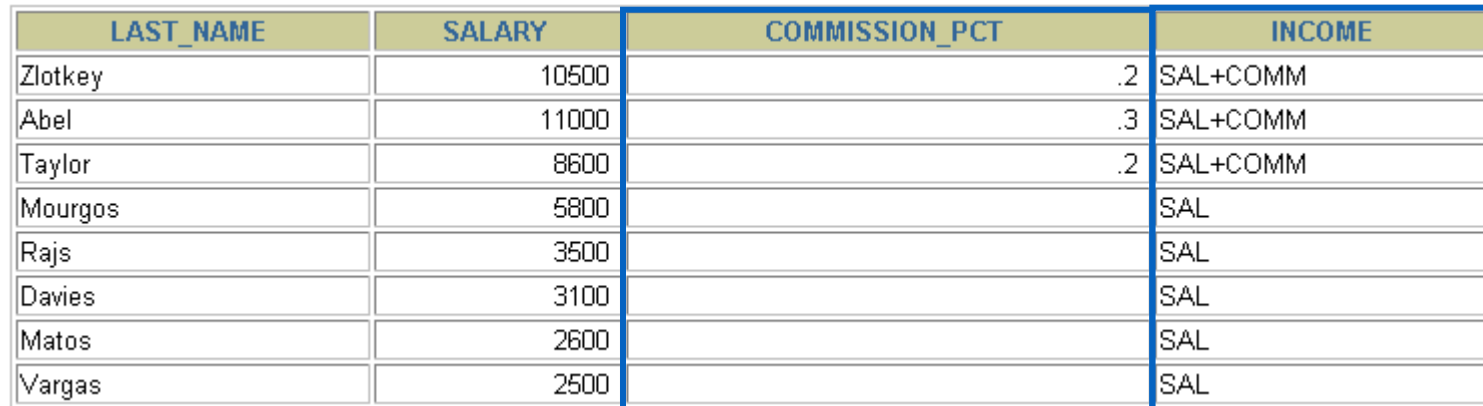
2

# Az NVL2 függvény használata

```
SELECT last name, salary, commission pct,
       NVL2 (commission_pct,
            'SAL+COMM', 'SAL') income
FROM   employees WHERE department_id IN (50, 80);
```



LAST_NAME	SALARY	COMMISSION_PCT	INCOME
Zlotkey	10500	.2	SAL+COMM
Abel	11000	.3	SAL+COMM
Taylor	8600	.2	SAL+COMM
Mourgos	5800		SAL
Rajs	3500		SAL
Davies	3100		SAL
Matos	2600		SAL
Vargas	2500		SAL



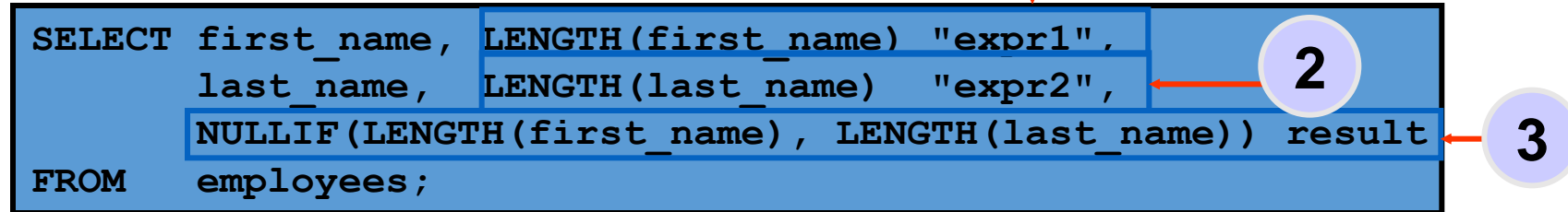
8 rows selected.

1

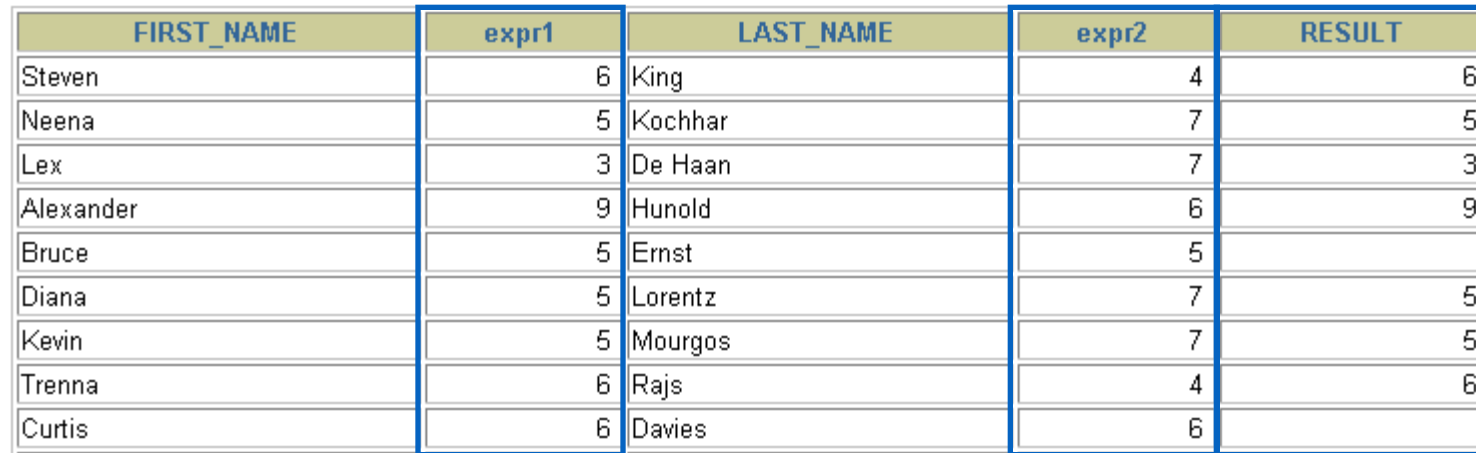
2

# Az NULLIF függvény használata

```
SELECT first_name, LENGTH(first_name) "expr1",  
       last_name, LENGTH(last_name) "expr2",  
       NULLIF(LENGTH(first_name), LENGTH(last_name)) result  
FROM employees;
```



FIRST_NAME	expr1	LAST_NAME	expr2	RESULT
Steven	6	King	4	6
Neena	5	Kochhar	7	5
Lex	3	De Haan	7	3
Alexander	9	Hunold	6	9
Bruce	5	Ernst	5	
Diana	5	Lorentz	7	5
Kevin	5	Mourgos	7	5
Trenna	6	Rajs	4	6
Curtis	6	Davies	6	



...  
20 rows selected.

1

2

3

# A COALESCE függvény használata

- A COALESCE függvény esetében - az NVL függvénnel szemben - több helyettesítő értéket is megadhatunk.
- Ha az első kifejezés nem nullértéket ad vissza, akkor ez a függvény értéke, különben a COALESCE függvényt alkalmazza a maradék kifejezésekre.

# A COALESCE függvény használata

```
SELECT last_name,  
       COALESCE(manager_id,commission_pct, -1) comm  
FROM   employees  
ORDER BY commission_pct;
```

LAST_NAME	COMM
Grant	149
Zlotkey	100
Taylor	149
Abel	149
King	-1
Kochhar	100
De Haan	100

\*\*\*

20 rows selected.



# Feltételes kifejezések

- Segítségükkel IF-THEN-ELSE típusú logikát lehet használni az SQL utasításban
- Kétféle módszert használhatunk:
  - CASE expression
  - DECODE function

# A CASE kifejezés

- Feltételes lekérdezéseket lehet megfogalmazni vele az IF-THEN-ELSE utasításhoz hasonlóan:

```
CASE expr WHEN comparison_expr1 THEN return_expr1  
      [WHEN comparison_expr2 THEN return_expr2  
      WHEN comparison_exprn THEN return_exprn  
      ELSE else_expr]  
END
```

# A CASE kifejezés használata

```
SELECT last_name, job_id, salary,  
       CASE job_id WHEN 'IT_PROG' THEN 1.10*salary  
                 WHEN 'ST_CLERK' THEN 1.15*salary  
                 WHEN 'SA_REP' THEN 1.20*salary  
       ELSE salary END "REVISED_SALARY"  
FROM employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

# A DECODE függvény

Feltételes lekérdezéseket lehet megfogalmazni vele a CASE vagy az IF-THEN-ELSE utasításhoz hasonlóan:

```
DECODE(col|expression, search1, result1  
      [, search2, result2, ..., ]  
      [, default])
```

# A DECODE függvény használata

```
SELECT last name, job id, salary,  
       DECODE(job_id, 'IT_PROG', 1.10*salary,  
                'ST_CLERK', 1.15*salary,  
                'SA_REP', 1.20*salary,  
                salary)  
       REVISED_SALARY  
FROM   employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

# A DECODE függvény használata

```
SELECT last name, salary,  
       DECODE (TRUNC(salary/2000, 0),  
              0, 0.00,  
              1, 0.09,  
              2, 0.20,  
              3, 0.30,  
              4, 0.40,  
              5, 0.42,  
              6, 0.44,  
              0.45) TAX_RATE  
FROM   employees  
WHERE  department_id = 80;
```

# Összefoglalás

- Ebben a részben megtanultuk:
  - hogyan kell a dátumokkal műveleteket végezni, függvényekben dátumokat használni
  - hogyan lehet módosítani az adatokat függvények segítségével
  - hogyan lehet a lekérdezés eredményét adó sorokat formázni
  - hogyan lehet különböző dátumformátumokat használni a megjelenítésben
  - hogyan lehet az adattípusokat konvertálni
  - hogyan kell használni az NVL függvényt
  - hogyan működnek a feltételes - IF-THEN-ELSE – logikájú kifejezések