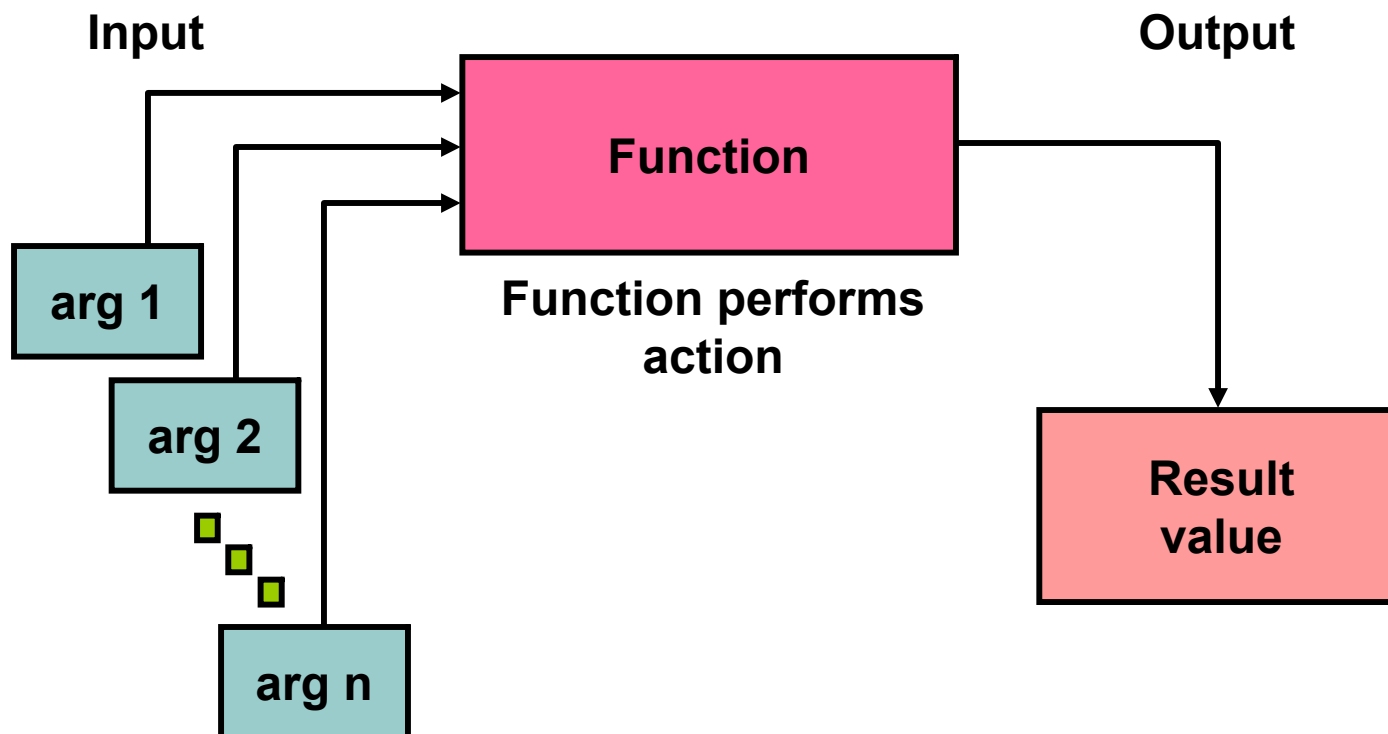# Using Single-Row Functions to Customize Output
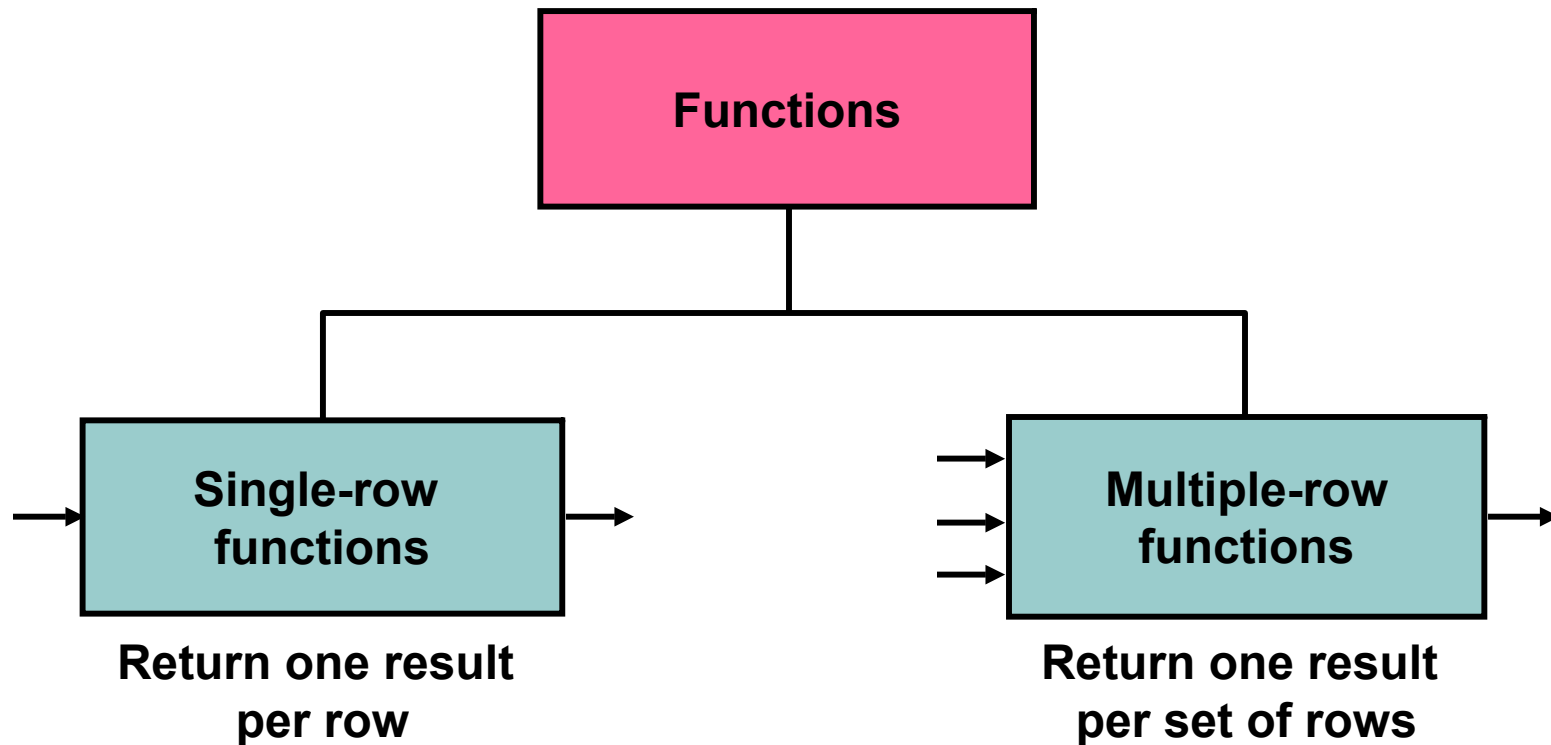
ORACLE

Objectives

After completing this lesson, you should be able to do the following:

- Describe various types of functions that are available in SQL

- Use character, number, and date functions in `SELECT` statements

- Describe the use of conversion functions

ORACLE

SQL Functions

Input   **Output**

Function
Function performs action

arg 1

arg 2

arg n

| Result value |

ORACLE

Two Types of SQL Functions

```
                    ┌──────────────────┐
                    │    Functions     │
                    └──────────────────┘
                             │
              ┌──────────────┴──────────────┐
    ┌──────────────────┐          ┌──────────────────┐
──▶ │    Single-row    │ ──▶   ──▶│   Multiple-row   │ ──▶
    │    functions     │      ──▶ │    functions     │
    └──────────────────┘      ──▶ └──────────────────┘

      Return one result          Return one result
          per row                  per set of rows
```
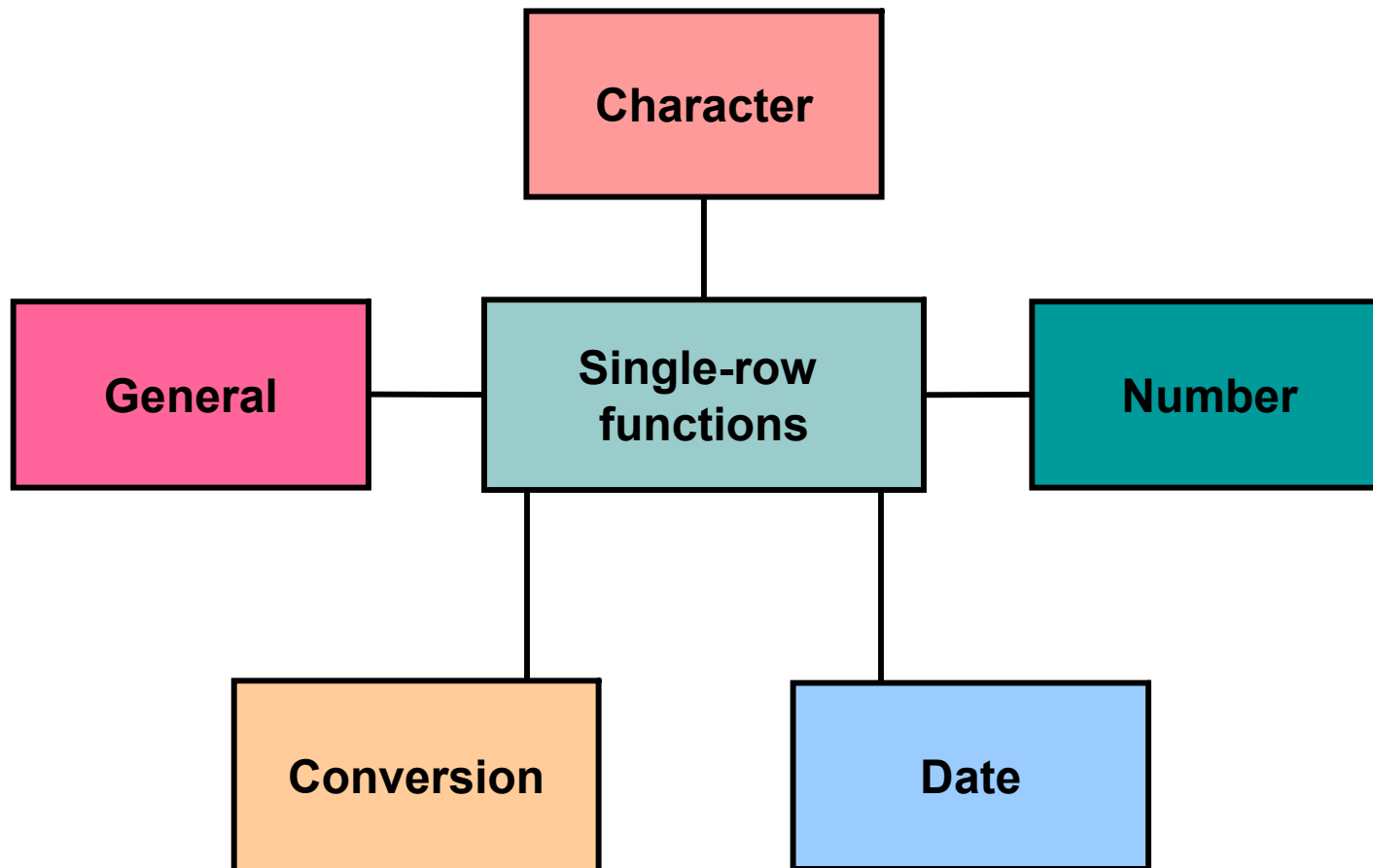
Single-Row Functions

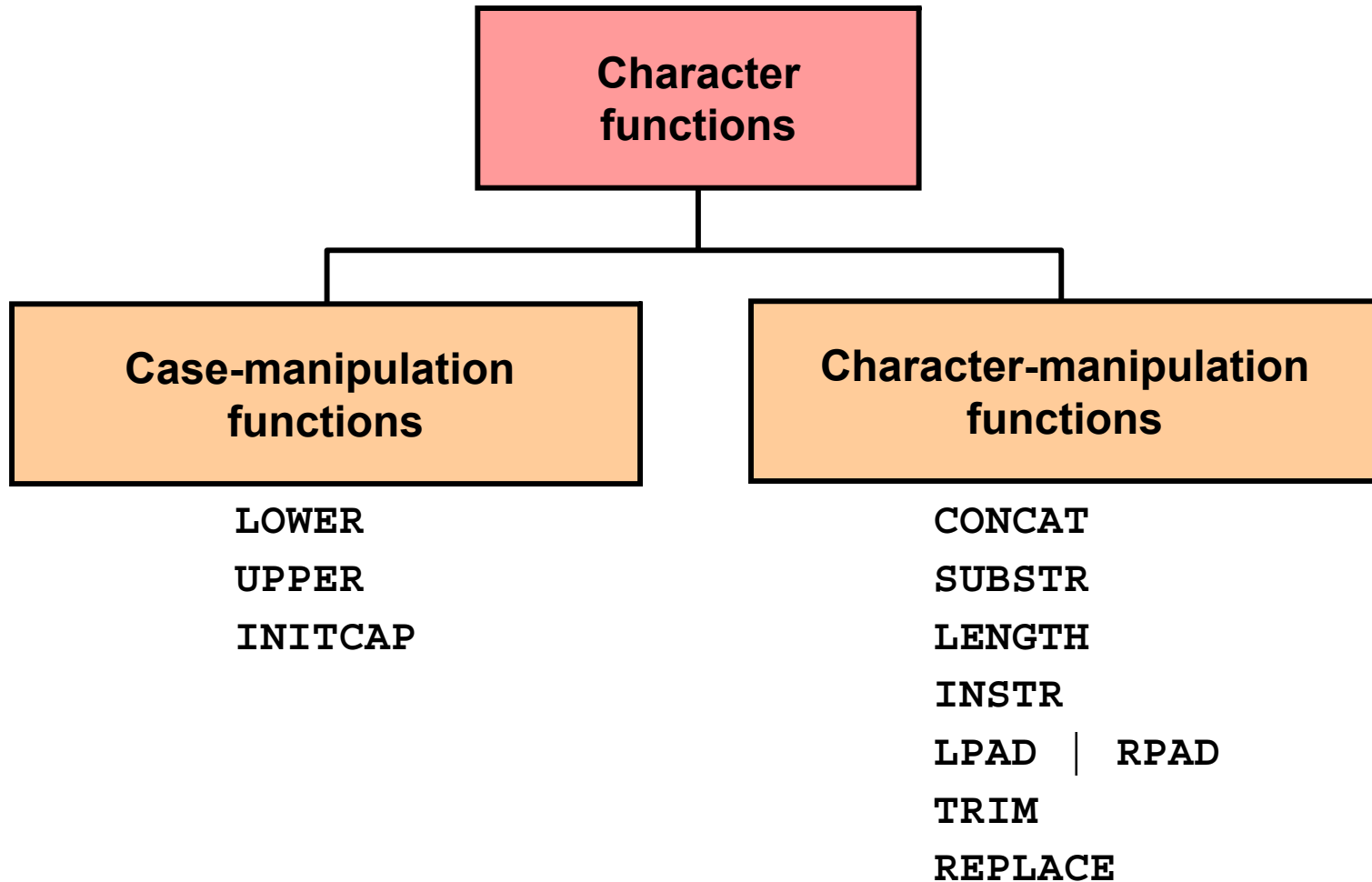**Single-row functions:**

- **Manipulate data items**

- **Accept arguments and return one value**

- **Act on each row that is returned**

- **Return one result per row**

- **May modify the data type**

- **Can be nested**

- **Accept arguments that can be a column or an expression**

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

ORACLE

Single-Row Functions

ORACLE

Character Functions

```
                    ┌─────────────────┐
                    │    Character    │
                    │    functions    │
                    └─────────────────┘
              ┌─────────────┴─────────────┐
┌──────────────────────┐      ┌──────────────────────────┐
│  Case-manipulation   │      │  Character-manipulation  │
│     functions        │      │        functions         │
└──────────────────────┘      └──────────────────────────┘

LOWER                          CONCAT

UPPER                          SUBSTR

INITCAP                        LENGTH

                               INSTR

                               LPAD │ RPAD

                               TRIM

                               REPLACE
```

ORACLE

Case-Manipulation Functions

These functions convert case for character strings:

Function	Result
`LOWER('SQL Course')`	`sql course`
`UPPER('SQL Course')`	`SQL COURSE`
`INITCAP('SQL Course')`	`Sql Course`

ORACLE

Using Case-Manipulation Functions

Display the employee number, name, and department number for employee Higgins:

```
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';
```

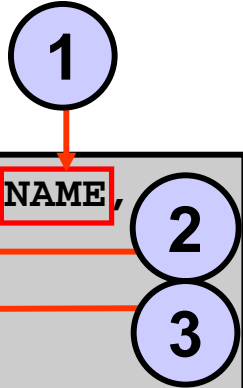| EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
205	Higgins	110

Character-Manipulation Functions

These functions manipulate character strings:

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

ORACLE

Using the Character-Manipulation Functions

```
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';
```

1

2

3

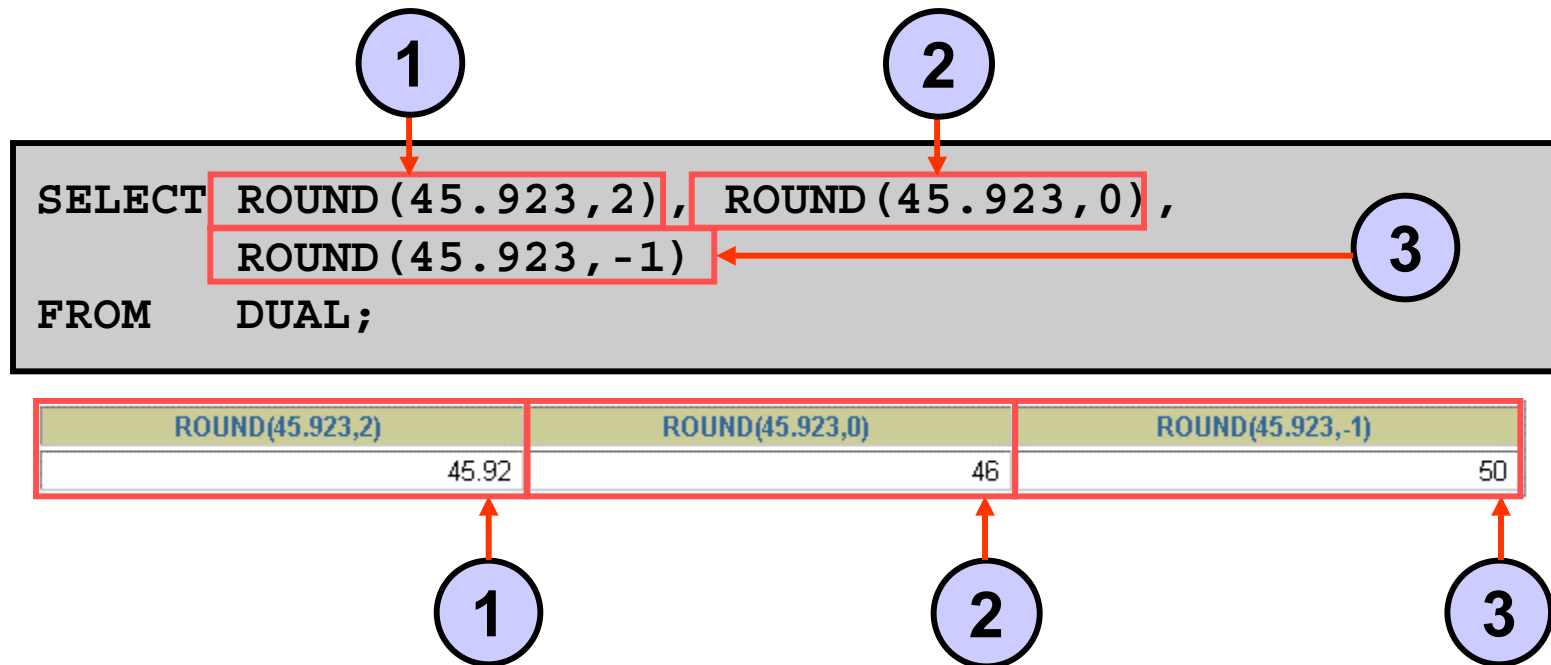| EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
174	EllenAbel	SA_REP	4	0
176	JonathonTaylor	SA_REP	6	2
178	KimberelyGrant	SA_REP	5	3
202	PatFay	MK_REP	3	2

1   **2**   **3**

ORACLE

Number Functions

- **ROUND: Rounds value to specified decimal**
- **TRUNC: Truncates value to specified decimal**
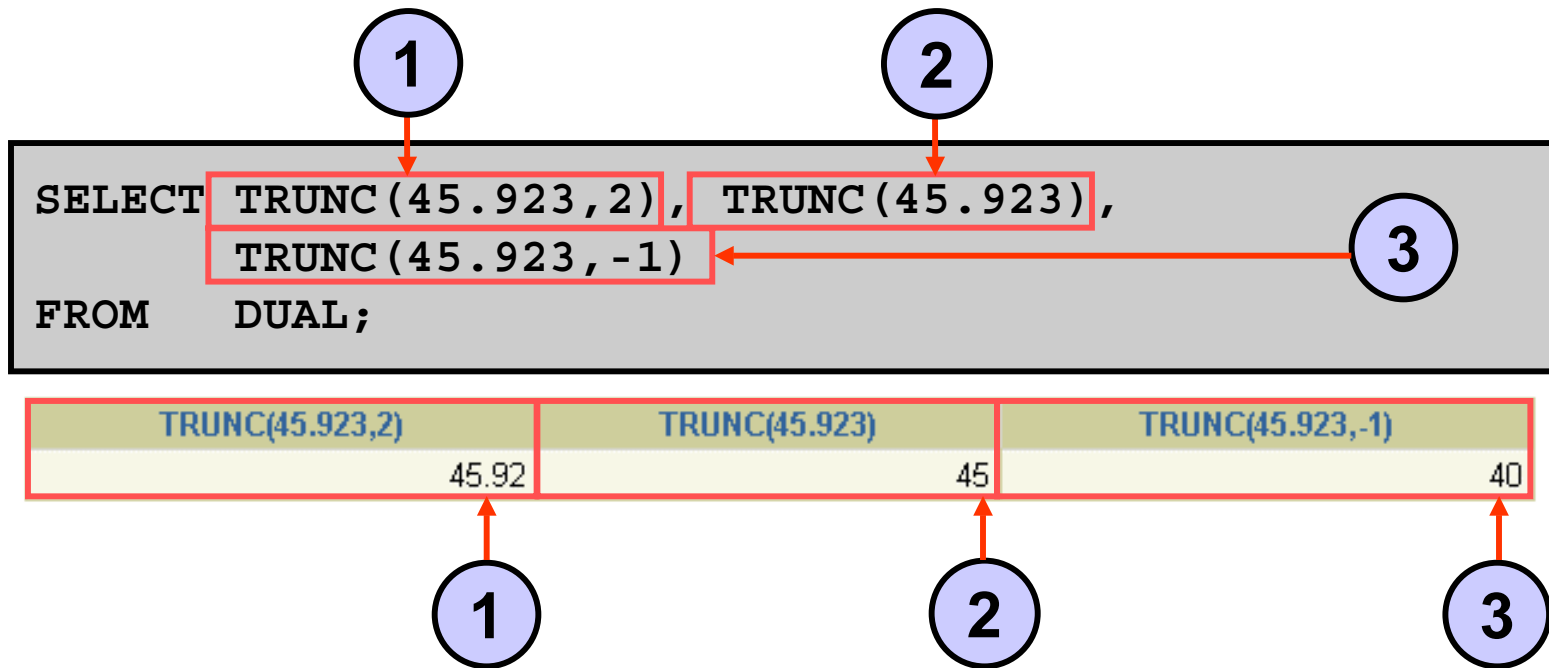- **MOD: Returns remainder of division**

Function	Result
ROUND(45.926, 2)	45.93
TRUNC(45.926, 2)	45.92
MOD(1600, 300)	100

ORACLE

Using the ROUND Function

SELECT ROUND(45.923,2), ROUND(45.923,0),
       ROUND(45.923,-1)
FROM   DUAL;

ROUND(45.923,2)	ROUND(45.923,0)	ROUND(45.923,-1)
45.92	46	50

DUAL is a dummy table that you can use to view results from functions and calculations.

ORACLE

Using the TRUNC Function


```
          1                    2
SELECT  TRUNC(45.923,2),   TRUNC(45.923),
        TRUNC(45.923,-1)                        3
FROM    DUAL;
```

TRUNC(45.923,2)	TRUNC(45.923)	TRUNC(45.923,-1)
45.92	45	40

ORACLE

Using the MOD Function

For all employees with job title of Sales Representative, calculate the remainder of the salary after it is divided by 5,000.

```
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

ORACLE

Working with Dates

- **The Oracle database stores dates in an internal numeric format: century, year, month, day, hours, minutes, and seconds.**
- **The default date display format is DD-MON-RR.**
  - **Enables you to store 21st-century dates in the 20th century by specifying only the last two digits of the year**
  - **Enables you to store 20th-century dates in the 21st century in the same way**

```
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

ORACLE

Working with Dates

`SYSDATE` is a function that returns:

- Date
- Time

ORACLE

Arithmetic with Dates

- **Add or subtract a number to or from a date for a resultant date value.**

- **Subtract two dates to find the number of days between those dates.**

- **Add hours to a date by dividing the number of hours by 24.**

ORACLE

Using Arithmetic Operators
with Dates

```
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

ORACLE

Date Functions

Function	Result
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Next day of the date specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

Using Date Functions

Function	Result
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'

ORACLE

Using Date Functions

Assume `SYSDATE = '25-JUL-03':`

Function	Result
`ROUND(SYSDATE,'MONTH')`	`01-AUG-03`
`ROUND(SYSDATE ,'YEAR')`	`01-JAN-04`
`TRUNC(SYSDATE ,'MONTH')`	`01-JUL-03`
`TRUNC(SYSDATE ,'YEAR')`	`01-JAN-03`

ORACLE

Practice 3: Overview of Part 1

This practice covers the following topics:

- Writing a query that displays the current date
- Creating queries that require the use of numeric, character, and date functions
- Performing calculations of years and months of service for an employee

ORACLE

Conversion Functions
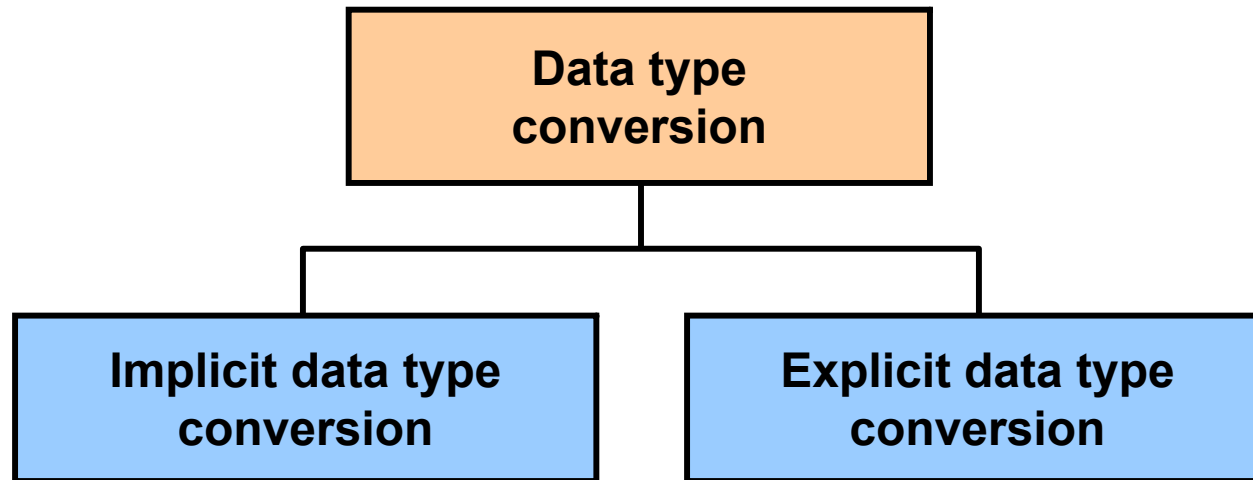
ORACLE

Implicit Data Type Conversion

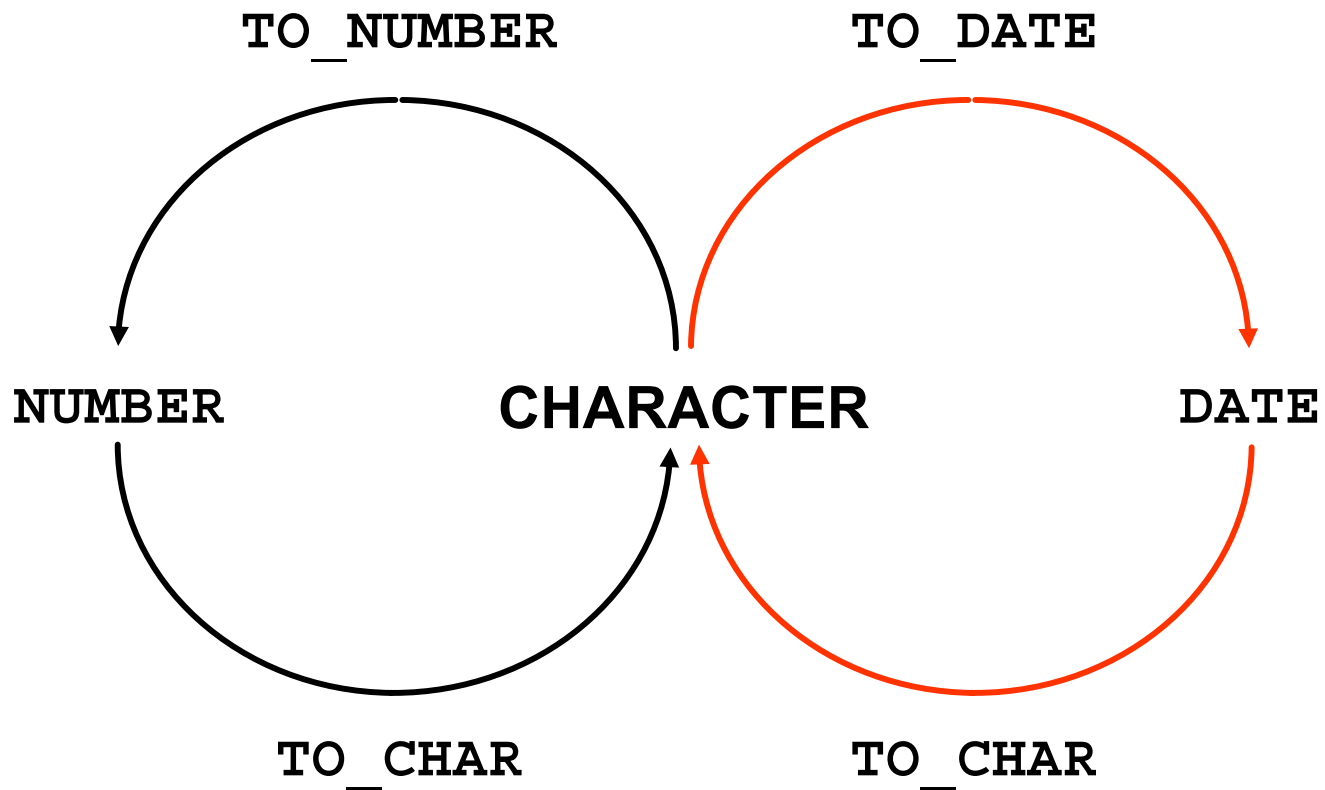**For assignments, the Oracle server can automatically convert the following:**

From	To
`VARCHAR2 or CHAR`	`NUMBER`
`VARCHAR2 or CHAR`	`DATE`
`NUMBER`	`VARCHAR2`
`DATE`	`VARCHAR2`

ORACLE

Implicit Data Type Conversion

For expression evaluation, the Oracle Server can automatically convert the following:

From	To
`VARCHAR2 or CHAR`	`NUMBER`
`VARCHAR2 or CHAR`	`DATE`

ORACLE

Explicit Data Type Conversion

TO_NUMBER   TO_DATE

NUMBER   CHARACTER   DATE

TO_CHAR   TO_CHAR

ORACLE

Explicit Data Type Conversion

TO_NUMBER   TO_DATE

NUMBER   **CHARACTER**   DATE

TO_CHAR   TO_CHAR

ORACLE
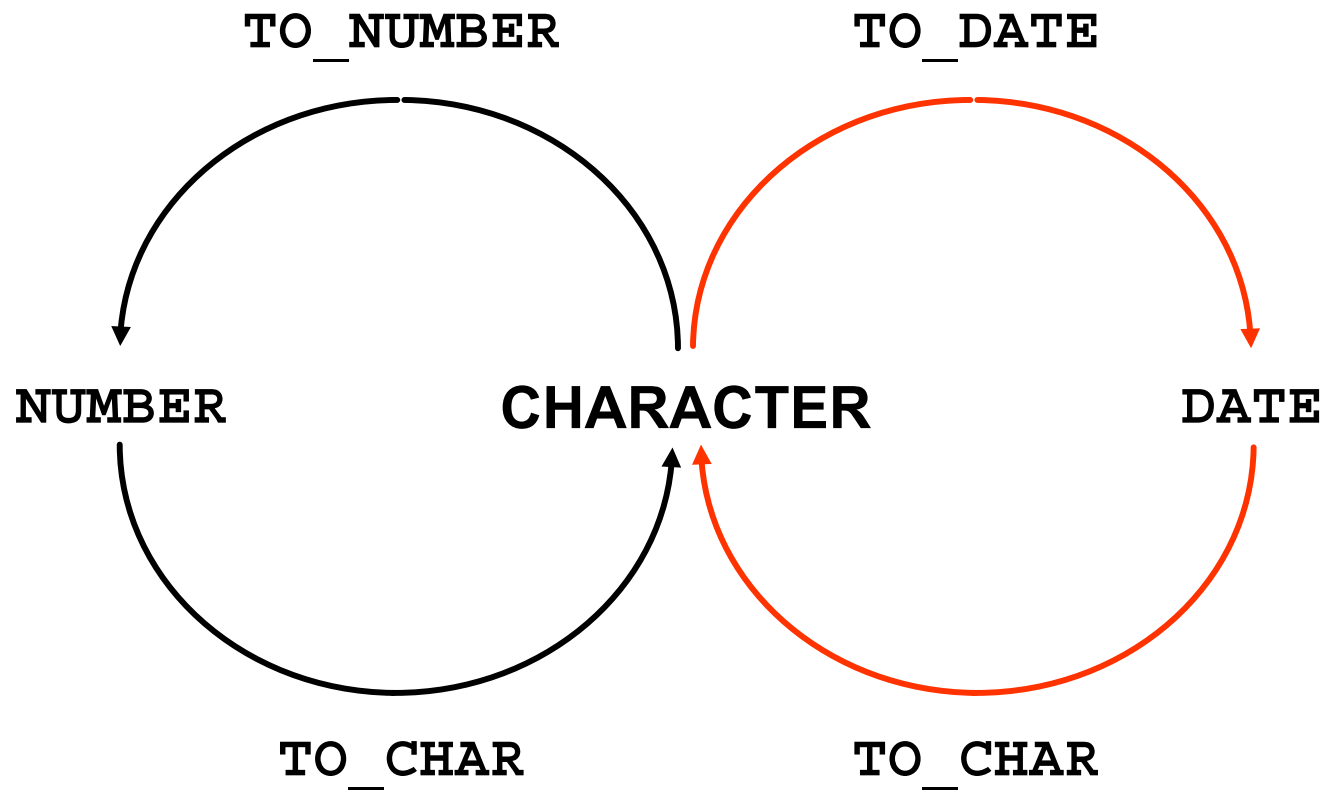
Using the `TO_CHAR` Function with Dates

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

The format model:

- Must be enclosed by single quotation marks
- Is case sensitive
- Can include any valid date format element
- Has an `fm` element to remove padded blanks or suppress leading zeros
- Is separated from the date value by a comma

ORACLE

Elements of the Date Format Model

Element	Result
YYYY	Full year in numbers
YEAR	Year spelled out (in English)
MM	Two-digit value for month
MONTH	Full name of the month
MON	Three-letter abbreviation of the month
DY	Three-letter abbreviation of the day of the week
DAY	Full name of the day of the week
DD	Numeric day of the month

ORACLE

Elements of the Date Format Model

- **Time elements format the time portion of the date:**

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

- **Add character strings by enclosing them in double quotation marks:**

DD "of" MONTH	12 of OCTOBER

- **Number suffixes spell out numbers:**

ddspth	fourteenth

ORACLE

Using the `TO_CHAR` Function with Dates

```
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

...

20 rows selected.

ORACLE

Using the `TO_CHAR` Function with Numbers

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

These are some of the format elements that you can use with the `TO_CHAR` function to display a number value as a character:

Element	Result
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 comma as thousands indicator

ORACLE

Using the `TO_CHAR` Function with Numbers

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

SALARY
$6,000.00

ORACLE

Using the `TO_NUMBER` and `TO_DATE` Functions

- **Convert a character string to a number format using the `TO_NUMBER` function:**

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

- **Convert a character string to a date format using the `TO_DATE` function:**

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

- **These functions have an `fx` modifier. This modifier specifies the exact matching for the character argument and date format model of a `TO_DATE` function.**

RR Date Format

Current Year	Specified Date	RR Format	YY Format
1995	27-OCT-95	1995	1995
1995	27-OCT-17	2017	1917
2001	27-OCT-17	2017	2017
2001	27-OCT-95	1995	2095

		If the specified two-digit year is:	
		0–49	50–99
If two digits of the current year are:	0–49	The return date is in the current century	The return date is in the century before the current one
	50–99	The return date is in the century after the current one	The return date is in the current century

ORACLE

Example of RR Date Format

To find employees hired before 1990, use the RR date format, which produces the same results whether the command is run in 1999 or now:

```
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

ORACLE

Nesting Functions

- **Single-row functions can be nested to any level.**

- **Nested functions are evaluated from the deepest level to the least deep level.**

```
F3(F2(F1(col,arg1),arg2),arg3)
```

Step 1 = Result 1

Step 2 = Result 2

Step 3 = Result 3

ORACLE
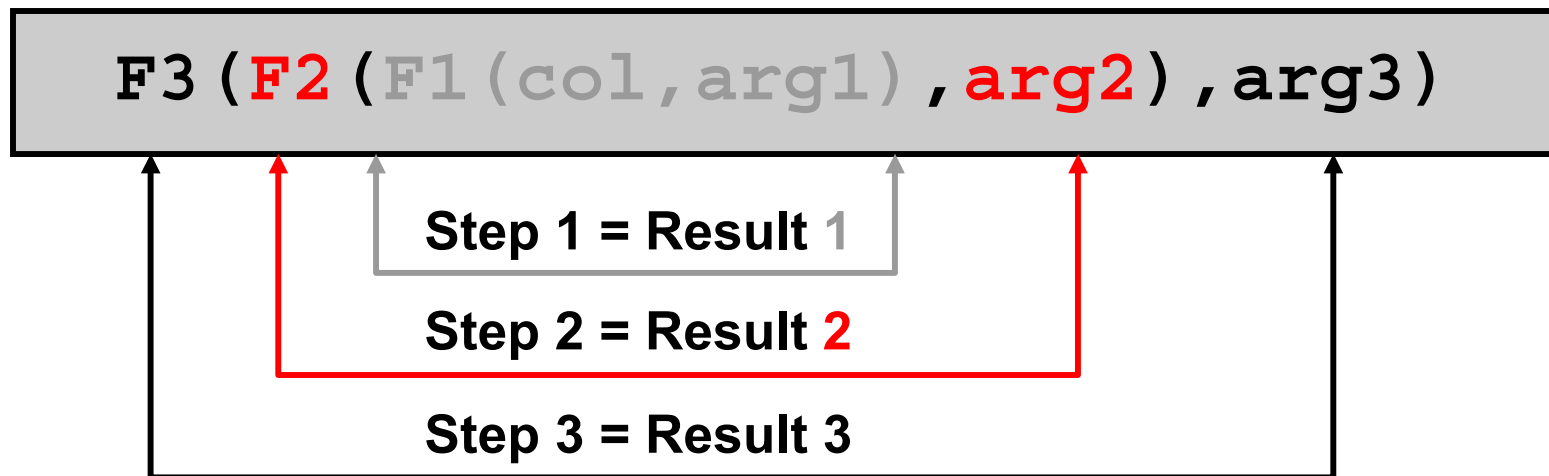
Nesting Functions

```
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

ORACLE

General Functions

The following functions work with any data type and pertain to using nulls:

- `NVL (expr1, expr2)`

- `NVL2 (expr1, expr2, expr3)`

- `NULLIF (expr1, expr2)`

- `COALESCE (expr1, expr2, ..., exprn)`

ORACLE

`NVL` Function

Converts a null value to an actual value:

- Data types that can be used are date, character, and number.

- Data types must match:
  - `NVL(commission_pct,0)`
  - `NVL(hire_date,'01-JAN-97')`
  - `NVL(job_id,'No Job Yet')`

ORACLE

Using the NVL Function

```
SELECT last_name, salary, NVL(commission_pct, 0),
    (salary*12) + (salary*12*NVL(commission_pct, 0)) AN_SAL
FROM employees;
```
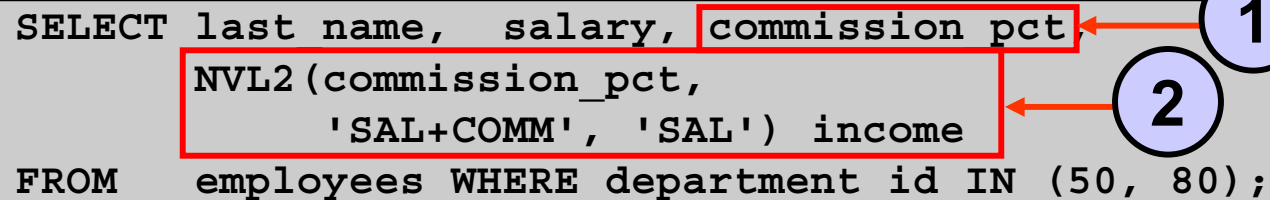
LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
King	24000	0	288000
Kochhar	17000	0	204000
De Haan	17000	0	204000
Hunold	9000	0	108000
Ernst	6000	0	72000
Lorentz	4200	0	50400
Mourgos	5800	0	69600
Rajs	3500	0	42000

...
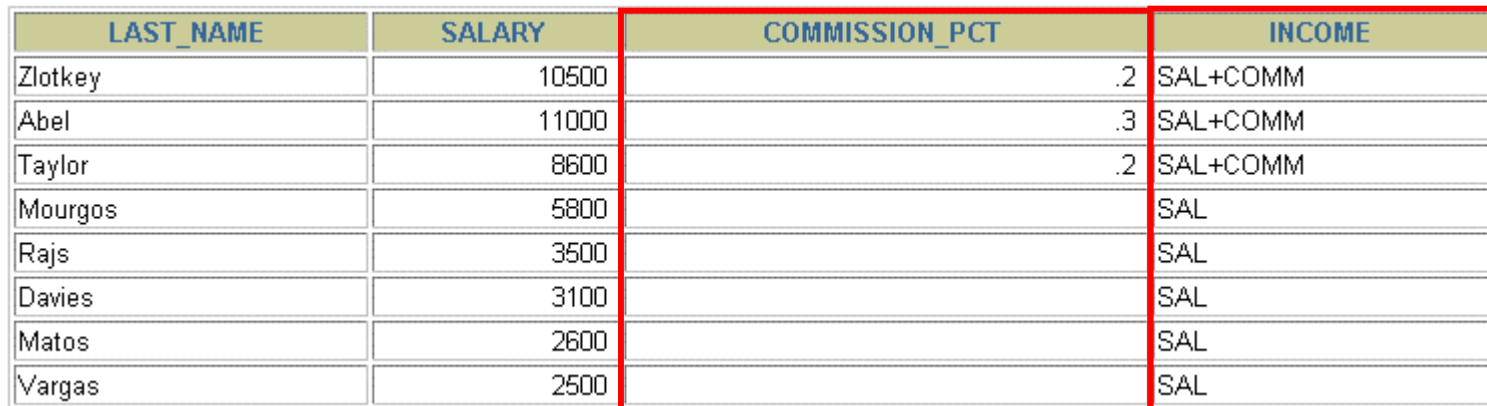
20 rows selected.

ORACLE

Using the NVL2 Function

```
SELECT last_name,  salary, commission_pct    ①
       NVL2(commission_pct,                   ②
            'SAL+COMM', 'SAL') income
FROM    employees WHERE department_id IN (50, 80);
```
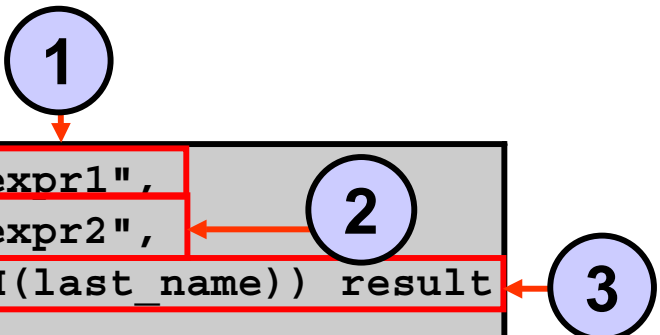
LAST_NAME	SALARY	COMMISSION_PCT	INCOME
Zlotkey	10500	.2	SAL+COMM
Abel	11000	.3	SAL+COMM
Taylor	8600	.2	SAL+COMM
Mourgos	5800		SAL
Rajs	3500		SAL
Davies	3100		SAL
Matos	2600		SAL
Vargas	2500		SAL

8 rows selected.

ORACLE

Using the `NULLIF` Function

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

1 **2** **3**

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**

ORACLE

Using the `COALESCE` Function

- **The advantage of the `COALESCE` function over the `NVL` function is that the `COALESCE` function can take multiple alternate values.**

- **If the first expression is not null, the `COALESCE` function returns that expression; otherwise, it does a `COALESCE` of the remaining expressions.**

ORACLE

Using the COALESCE Function

```
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.

ORACLE

Conditional Expressions

- **Provide the use of IF-THEN-ELSE logic within a SQL statement**

- **Use two methods:**
  - `CASE` **expression**
  - `DECODE` **function**

ORACLE

`CASE` Expression

Facilitates conditional inquiries by doing the work of an IF-THEN-ELSE statement:

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

ORACLE

Using the `CASE` Expression

Facilitates conditional inquiries by doing the work of an IF-THEN-ELSE statement:

```
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.

ORACLE

`DECODE` Function

Facilitates conditional inquiries by doing the work of a `CASE` expression or an IF-THEN-ELSE statement:

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

ORACLE

Using the DECODE Function

```
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.

ORACLE

Using the DECODE Function

Display the applicable tax rate for each employee in department 80:

```
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;
```

ORACLE

Summary

In this lesson, you should have learned how to:

- Perform calculations on data using functions

- Modify individual data items using functions

- Manipulate output for groups of rows using functions

- Alter date formats for display using functions

- Convert column data types using functions

- Use `NVL` functions

- Use IF-THEN-ELSE logic

ORACLE

Practice 3: Overview of Part 2

This practice covers the following topics:

- Creating queries that require the use of numeric, character, and date functions

- Using concatenation with functions

- Writing non-case-sensitive queries to test the usefulness of character functions

- Performing calculations of years and months of service for an employee

- Determining the review date for an employee

ORACLE