

Using Conversion Functions and Conditional Expressions

Objectives

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

- Describe the various types of conversion functions that are available in SQL
- Use the TO_CHAR, TO_NUMBER, and TO_DATE conversion functions
- Apply conditional expressions in a SELECT statement



Course RoadMap

Lesson 1: Introduction

**Unit 1: Retrieving, Restricting,
and Sorting Data**

Unit 2: Joins, Subqueries, and
Set Operators

Unit 3: DML and DDL

▶ Lesson 2: Retrieving Data using SQL SELECT

▶ Lesson 3: Restricting and Sorting Data

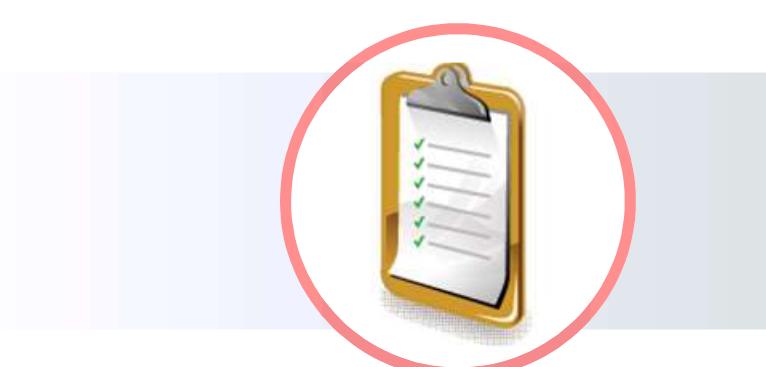
▶ Lesson 4: Using Single-Row Functions to
Customize Output

▶ **Lesson 5: Using Conversion Functions
and Conditional Expressions**

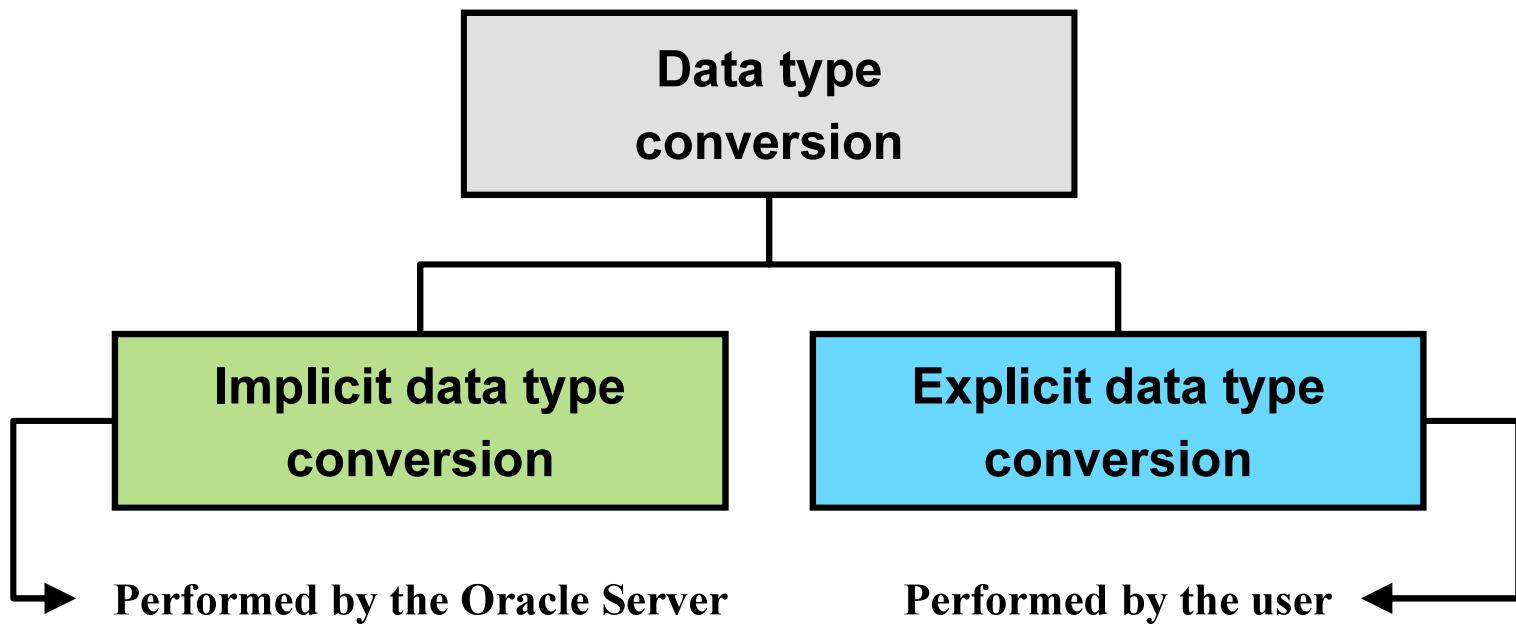
← You are here!

Lesson Agenda

- Implicit and explicit data type conversion
- TO_CHAR, TO_DATE, TO_NUMBER functions
- General functions:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Conditional expressions:
 - CASE
 - Searched CASE
 - DECODE



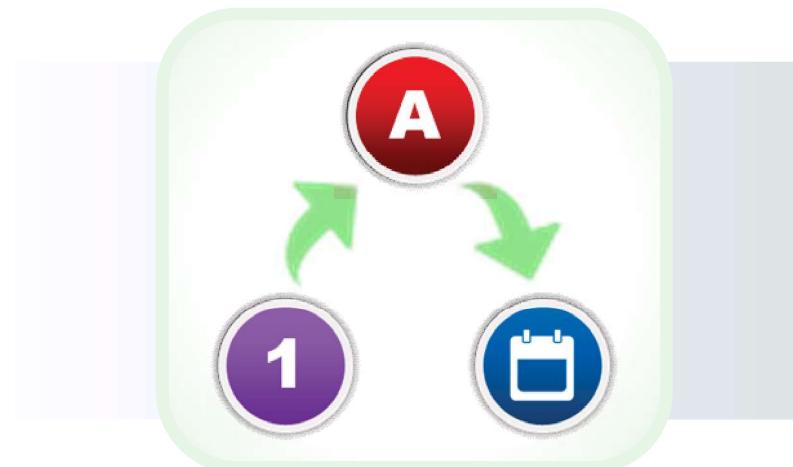
Conversion Functions



Implicit Data Type Conversion of Strings

- In expressions, the Oracle server can automatically convert the following:

From	To
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE

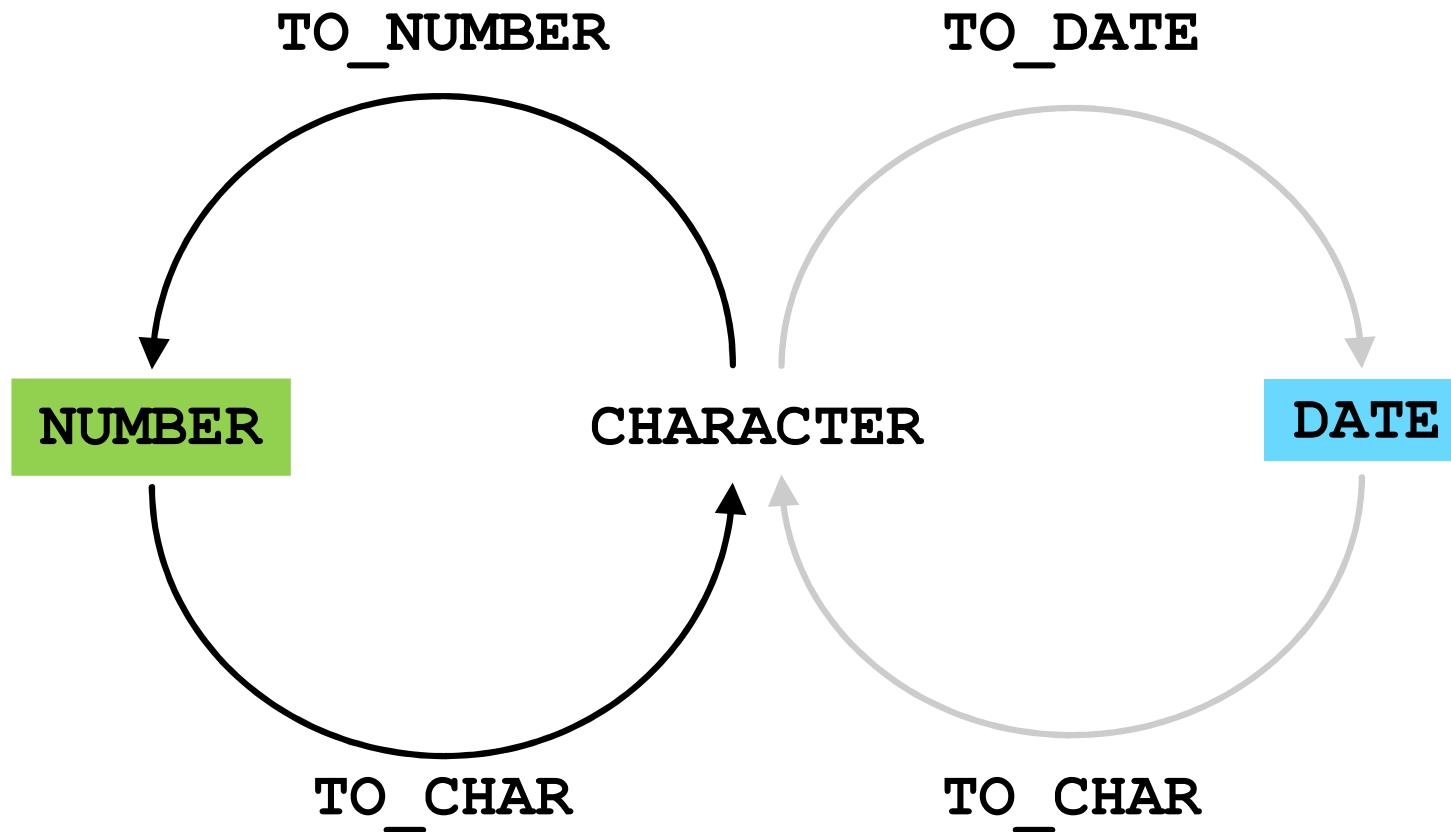


Implicit Data Type Conversion to Strings

- For expression evaluation, the Oracle server can automatically convert the following:

From	To
NUMBER	VARCHAR2 or CHAR
DATE	VARCHAR2 or CHAR

Explicit Data Type Conversion



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

Using the TO_CHAR Function with Dates

Example:

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

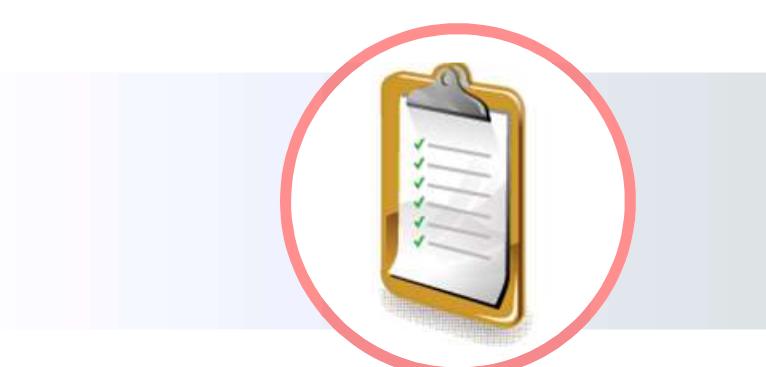
```
SELECT employee_id, TO_CHAR(hire_date, 'MM/YY')  
      Month_Hired  
  FROM employees  
 WHERE last_name = 'Higgins';
```

EMPLOYEE_ID	MONTH_HIRED
1	205 06/10



Lesson Agenda

- Implicit and explicit data type conversion
- TO_CHAR, TO_DATE, TO_NUMBER **functions**
- General functions:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Conditional expressions:
 - CASE
 - Searched CASE
 - DECODE



Elements of the Date Format Model

Element	Result
YYYY	Full year in numbers
YEAR	Year spelled out (in English)
MM	Two-digit value for the 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

Elements of the Date Format Model

- Time elements help you format the time portion of the date:

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

- Add character strings by enclosing them within double quotation marks:

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

- Number suffixes help in spelling out numbers:

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

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 2011
Kochhar	21 September 2009
De Haan	13 January 2009
Hunold	3 January 2014
Ernst	21 May 2015
Lorentz	7 February 2015
Mourgos	16 November 2015
Rajs	17 October 2011

...



Using the TO_CHAR Function with Numbers

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

TO_CHAR(*number*[, '*format_model*'])

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 a thousands indicator

Using the TO_CHAR Function with Numbers

Let us look at an example:

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

	SALARY
1	\$6,000.00



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'])
```


Using TO_CHAR and TO_DATE Functions with the RR

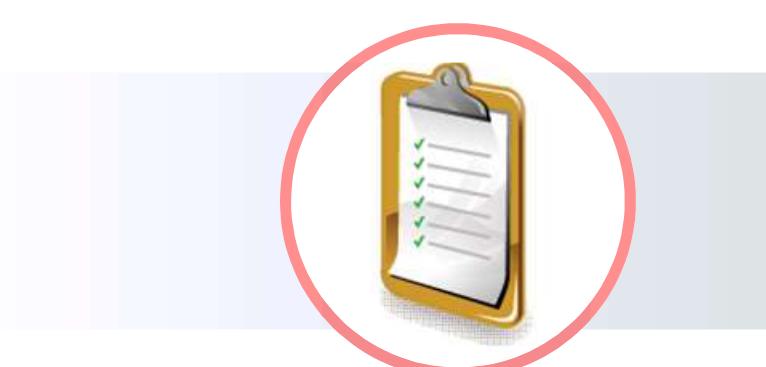
- To find employees hired before 2010, use the RR date format, which produces the correct result if the command is run now or before the year 2049:

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-YYYY')
FROM employees
WHERE hire_date < TO_DATE('01 Jan, 10', 'DD Mon,RR');
```

LAST_NAME	TO_CHAR(HIRE_DATE,'DD-MON-YYYY')
Kochhar	21-Sep-2009
De Haan	13-Jan-2009

Lesson Agenda

- Implicit and explicit data type conversion
- TO_CHAR, TO_DATE, TO_NUMBER functions
- General functions:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Conditional expressions:
 - CASE
 - Searched CASE
 - DECODE



General Functions

The following functions pertain to using nulls and can be used with any data type:

NVL (expr1, expr2)

NVL2 (expr1, expr2, expr3)

NULLIF (expr1, expr2)

**COALESCE (expr1, expr2,
..., exprn)**

1

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.
- Examples:
 - `NVL(commission_pct, 0)`
 - `NVL(hire_date, '01-JAN-97')`
 - `NVL(job_id, 'No Job Yet')`

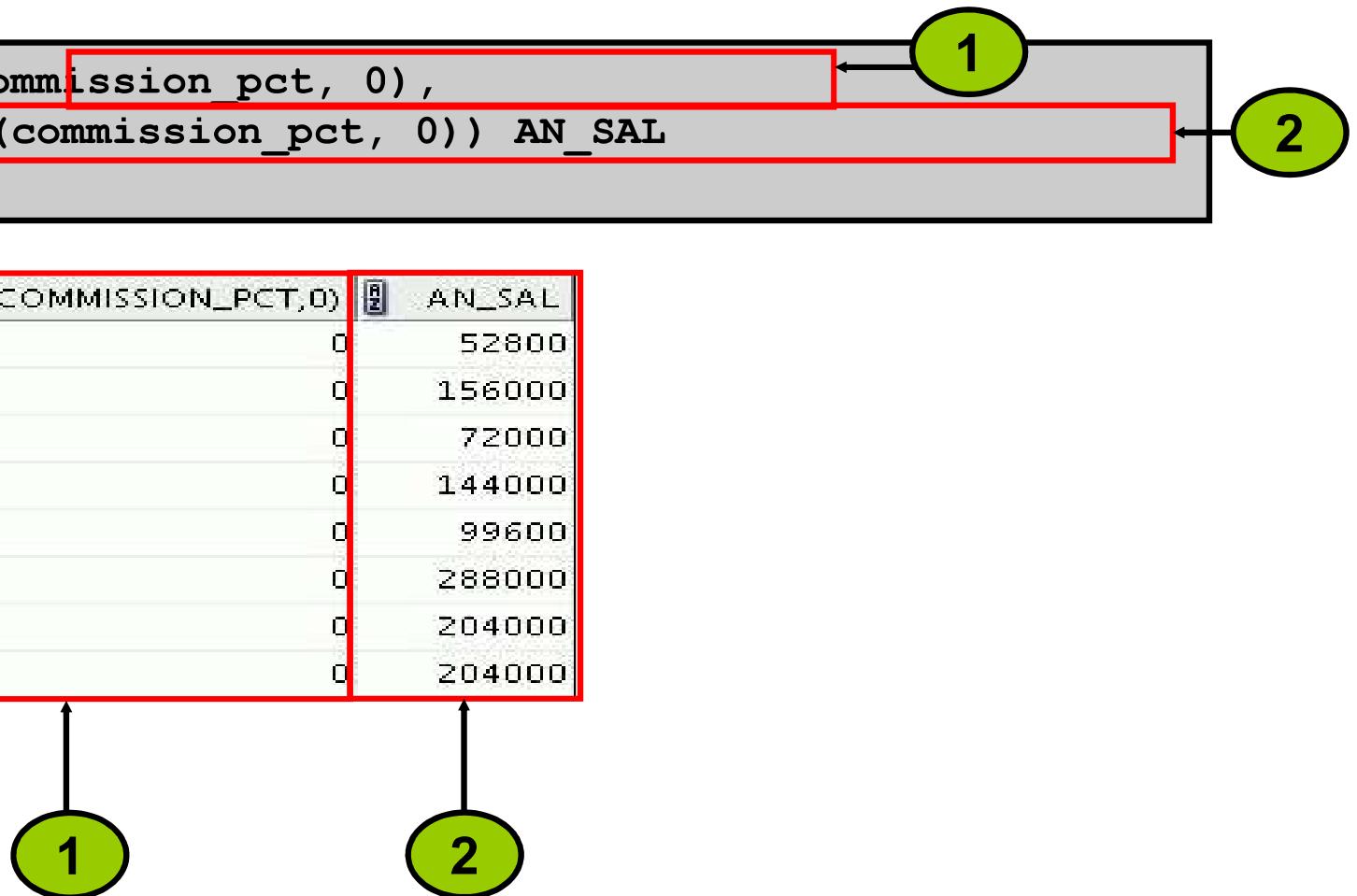
NVL (expr1, expr2)



Using the NVL Function

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

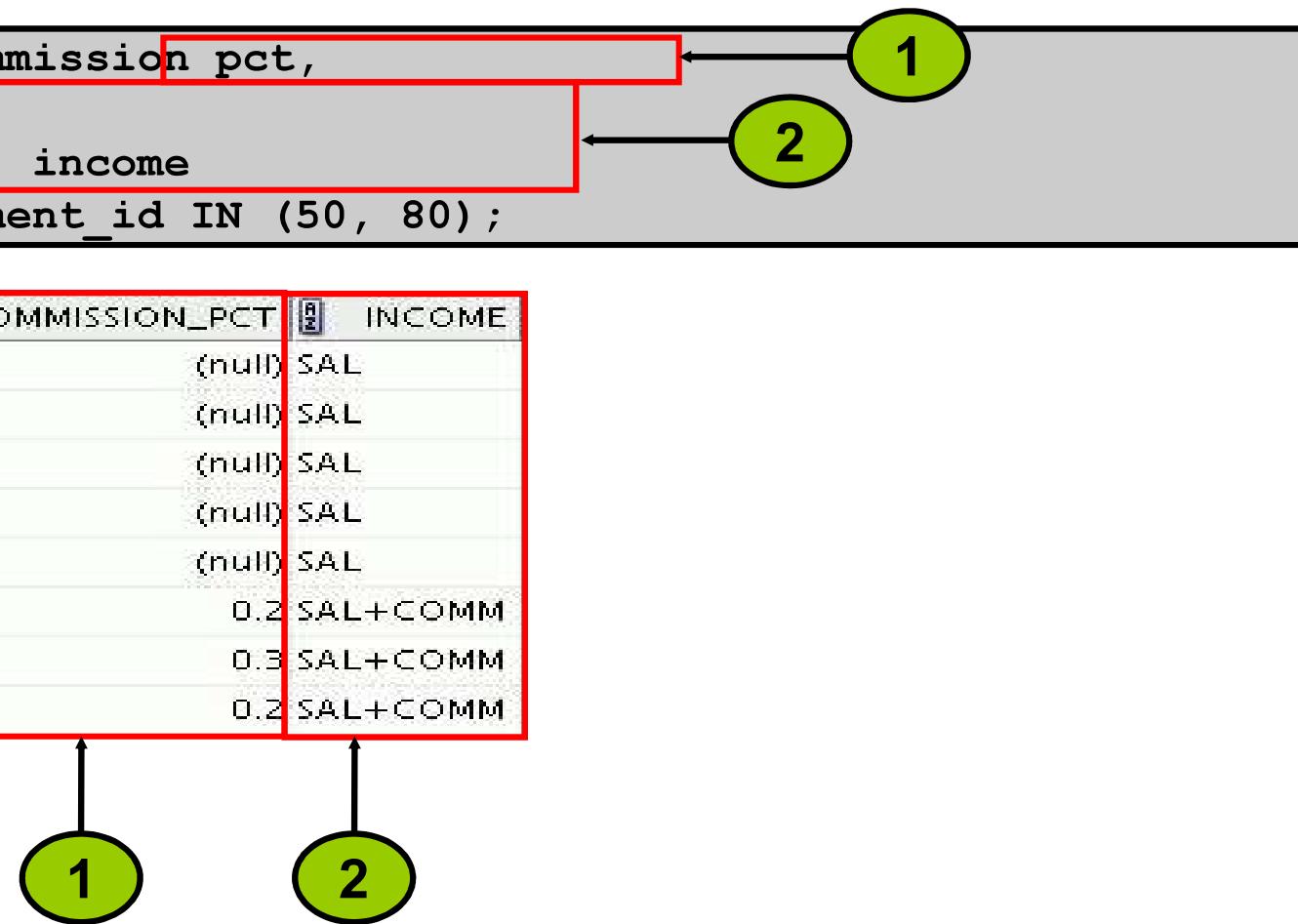
	LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
1	Whalen	4400	0	52800
2	Hartstein	13000	0	156000
3	Fay	6000	0	72000
4	Higgins	12000	0	144000
5	Gietz	8300	0	99600
6	King	24000	0	288000
7	Kochhar	17000	0	204000
8	De Haan	17000	0	204000
...				



Using the NVL2 Function

```
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
1	Mourgos	5800	(null)	SAL
2	Rajs	3500	(null)	SAL
3	Davies	3100	(null)	SAL
4	Matos	2600	(null)	SAL
5	Vargas	2500	(null)	SAL
6	Zlotkey	10500	0.2	SAL+COMM
7	Abel	11000	0.3	SAL+COMM
8	Taylor	8600	0.2	SAL+COMM

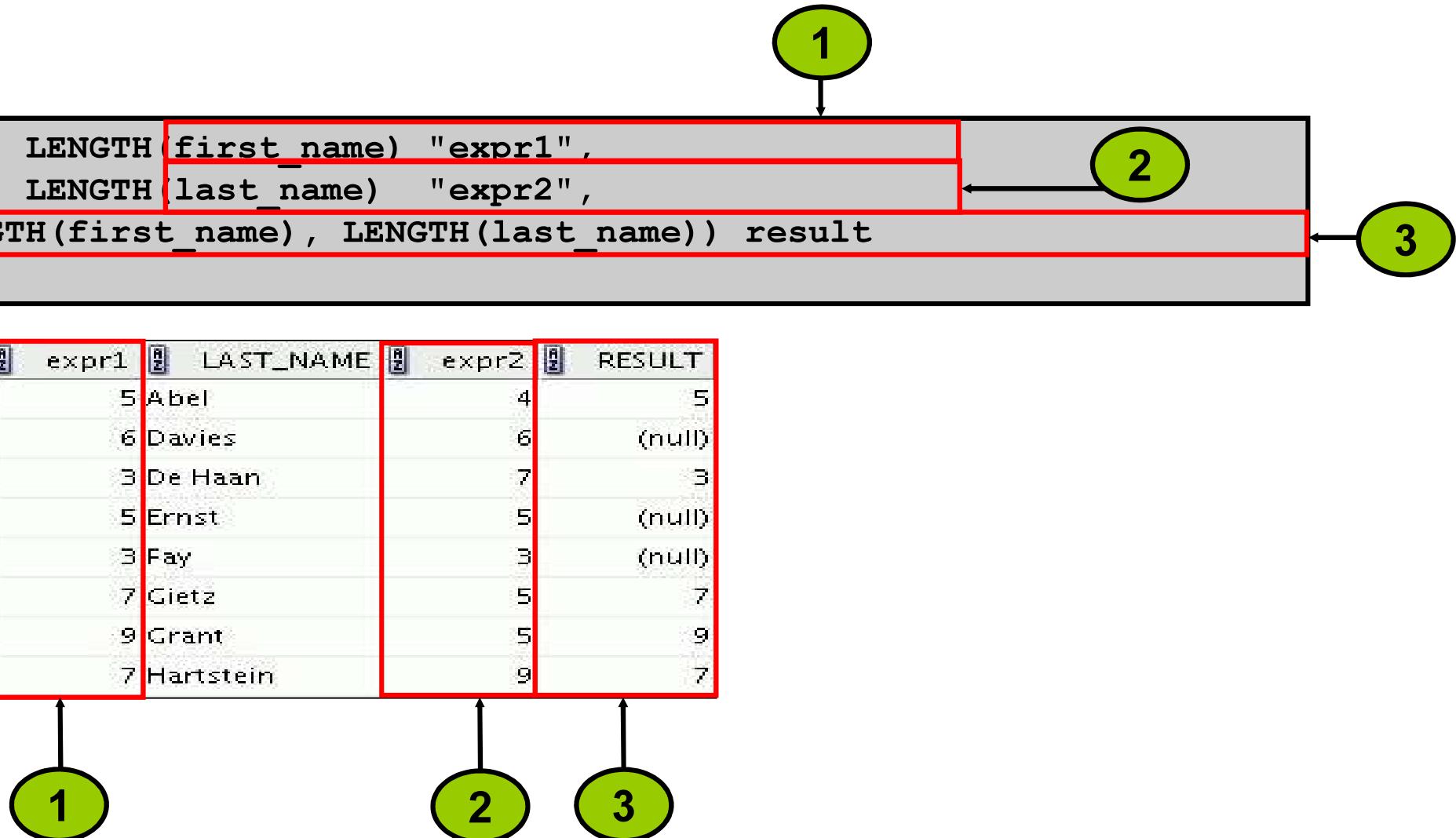


Using the NULLIF Function

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

	FIRST_NAME	expr1	LAST_NAME	expr2	RESULT
1	Ellen	5	Abel	4	5
2	Curtis		Davies	6	(null)
3	Lex		De Haan	7	3
4	Bruce		Ernst	5	(null)
5	Pat		Fay	3	(null)
6	William		Gietz	5	7
7	Kimberely		Grant	5	9
8	Michael		Hartstein	9	7

...



Using the COALESCE Function

- The advantage of the COALESCE function over the NVL function is that the COALESCE function can take multiple alternative values.
- If the first expression is not null, the COALESCE function returns that expression; otherwise, it does a COALESCE of the remaining expressions.

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

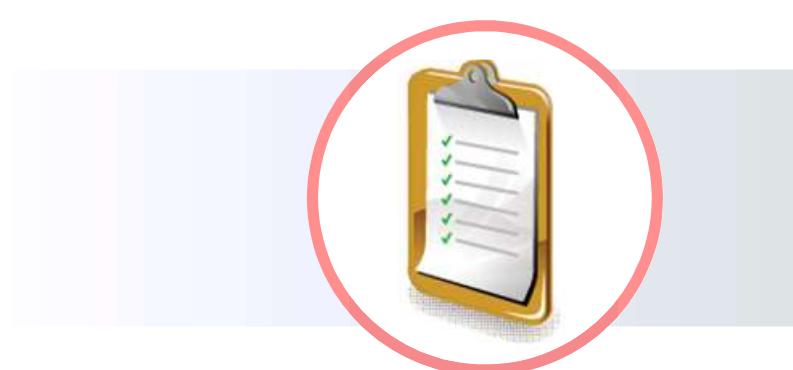
Using the COALESCE Function

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

	LAST_NAME	COMM
1	Grant	149
2	Taylor	149
3	Zlotkey	100
4	Abel	149
5	King	-1
6	Kochhar	100
7	De Haan	100
8	Hunold	102
...		

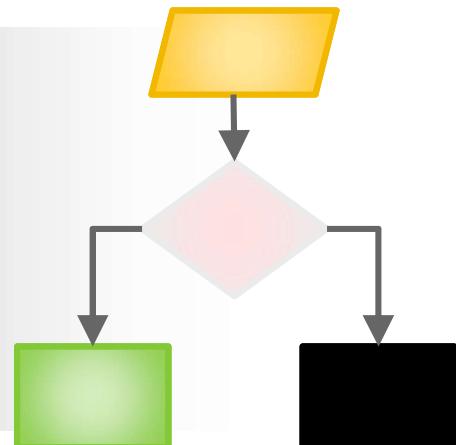
Lesson Agenda

- Implicit and explicit data type conversion
- TO_CHAR, TO_DATE, TO_NUMBER functions
- General functions:
 - NVL
 - NVL2
 - NULLIF
 - COALESCE
- Conditional expressions:
 - CASE
 - **Searched CASE**
 - DECODE



Conditional Expressions

- Help provide the use of IF-THEN-ELSE logic within a SQL statement
- You can use the following methods:
 - CASE expression
 - Searched CASE expression
 - DECODE function



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
```

Using the CASE Expression

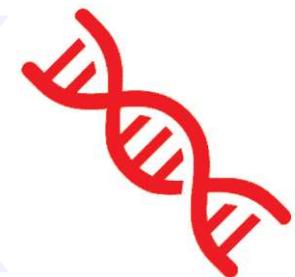
```
SELECT last_name, job_id, salary,  
       CASE job_id WHEN 'IT_PROG' THEN 1.10*salary  
                     WHEN 'ST_CLERK' THEN 1.15*salary  
                     WHEN 'SA REP' THEN 1.20*salary  
                     ELSE salary END "REVISED_SALARY"  
FROM employees;
```

	LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
1	King	AD_PRES	24000	24000
...				
4	Hunold	IT_PROG	9000	9900
5	Ernst	IT_PROG	6000	6600
6	Lorentz	IT_PROG	4200	4620
7	Mourgos	ST_MAN	5800	5800
8	Rajs	ST_CLERK	3500	4025
9	Davies	ST_CLERK	3100	3565
10	Matos	ST_CLERK	2600	2990
11	Vargas	ST_CLERK	2500	2875
...				
13	Abel	SA REP	11000	13200
14	Taylor	SA REP	8600	10320
15	Grant	SA REP	7000	8400

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])
```



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
...				
4	Hunold	IT_PROG	9000	9900
5	Ernst	IT_PROG	6000	6600
6	Lorentz	IT_PROG	4200	4620
7	Mourgos	ST_MAN	5800	5800
8	Rajs	ST_CLERK	3500	4025
9	Davies	ST_CLERK	3100	3565
10	Matos	ST_CLERK	2600	2990
11	Vargas	ST_CLERK	2500	2875
12	Zlotkey	SA_MAN	10500	10500
...				
13	Abel	SA REP	11000	13200
14	Taylor	SA REP	8600	10320
15	Grant	SA REP	7000	8400

Using the DECODE Function

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

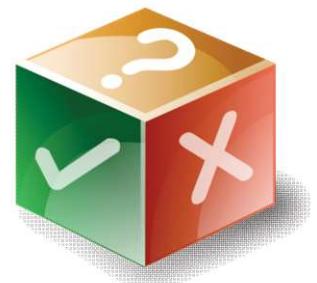
```
SELECT last_name, salary,  
       DECODE (TRUNC(salary/2000, 0),  
                0, 0.00,  
                1, 0.09,  
                2, 0.20,  
                3, 0.30,  
                4, 0.40,  
                5, 0.42,  
                6, 0.44,  
                0.45) TAX_RATE  
  FROM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

Quiz



The TO_NUMBER function converts either character strings or date values to a number in the format specified by the optional format model.

- a. True
- b. False



Summary

In this lesson, you should have learned how to:

- Alter date formats for display using functions
- Convert column data types using functions
- Use NVL functions
- Use IF-THEN-ELSE logic and other conditional expressions in a SELECT statement



Practice 3 – Part II : Overview

This practice covers the following topics:

- Creating queries that use TO_CHAR, TO_DATE, and other DATE functions
- Creating queries that use conditional expressions such as CASE, searched CASE, and DECODE

