

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

Academy

Database Programming with SQL

4-3

Date Functions

ORACLE
Academy



Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Objectives

This lesson covers the following objectives:

- Demonstrate the use of SYSDATE and date functions
- State the implications for world businesses to be able to easily manipulate data stored in date format

Purpose

- Have you ever wondered how many days remain in the school year or how many weeks there are until graduation?
- Because the Oracle database stores dates as numbers, you can easily perform calculations on dates using addition, subtraction, and other mathematical operators
- Businesses depend on being able to use date functions to schedule payrolls and payments, track employee performance reviews and years of service, or keep track of orders and shipments
- All of these business needs are easily handled using simple SQL date functions

Displaying Dates

- The default display and input format for dates is:
 - DD-Mon-YYYY
- For example: 02-Dec-2014
- However, the Oracle database stores dates internally with a numeric format representing the century, year, month, day, hour, minute, and second
- Valid Oracle dates are between January 1, 4712 B.C., and December 31, 9999 A.D.
- This represents the range of dates that you can store successfully in an Oracle database

SYSDATE

- SYSDATE is a date function that returns the current database server date and time
- Use SYSDATE to display the current date, use the DUAL table

```
SELECT SYSDATE  
FROM dual;
```

SYSDATE
01-Jul-2017

DATE Data Type

- The DATE data type always stores year information as a four-digit number internally: two digits for the century and two digits for the year
- For example, the Oracle database stores the year as 1996 or 2004, not just as 96 or 04
- In previous versions, the century component was not displayed by default
- However, due to changing business requirements around the world, the 4-digit year is now the default display

Working with Dates

Examples:	Description
<pre>SELECT last_name, hire_date + 60 FROM employees;</pre>	Adds 60 days to hire_date.
<pre>SELECT last_name, (SYSDATE - hire_date)/7 FROM employees;</pre>	Displays the number of weeks since the employee was hired.
<pre>SELECT employee_id, (end_date - start_date)/365 AS "Tenure in last job" FROM job_history;</pre>	Finds the number of days employee held a job, then divides by 365 to display in years.

These examples show the use of arithmetic operations with dates.

Date Functions

- The date functions shown in the table operate on Oracle dates
- All of the date functions return a value with a DATE data type except the MONTHS_BETWEEN function, which returns a numeric data type value

Function	Description
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Date of the next occurrence of day of the week specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

For the examples that follow, the SYSDATE was 01-Jul-2017

Date Functions

- **MONTHS_BETWEEN:** takes 2 DATE arguments and returns the number of calendar months between the 2 dates
- If the first argument is an earlier date than the second, the number returned is negative

Date Function Examples:	Result	
SELECT last_name, hire_date	King	17-Jun-1987
FROM employees	Kochhar	21-Sep-1989
WHERE MONTHS_BETWEEN	De Haan	13-Jan-1993
(SYSDATE, hire_date) > 240;

This example returns employees that were hired more than 240 months (20 years) ago.

Date Functions

- **ADD_MONTHS**: takes 2 arguments, a DATE and a number
- Returns a DATE value with the number argument added to the month component of the date
- If the number supplied is negative, the function will subtract that number of months from the date argument

Date Function Examples:	Result
<pre>SELECT ADD_MONTHS (SYSDATE, 12) AS "Next Year" FROM dual;</pre>	01-Jul-2016

This example adds 12 months to the current date.

Date Functions

- **NEXT_DAY**: takes 2 arguments, a **DATE** and a **weekday** and returns the **DATE** of the next occurrence of that weekday after the **DATE** argument

Date Function Examples:	Result
<pre>SELECT NEXT_DAY (SYSDATE, 'Saturday') AS "Next Saturday" FROM dual;</pre>	04-Jul-2017

Date Functions

- **LAST_DAY**: takes a DATE argument and returns the DATE of the last day of the month for the DATE argument

Date Function Examples:	Result
<pre>SELECT LAST_DAY (SYSDATE) AS "End of the Month" FROM dual;</pre>	31-Jul-2017

Date Functions

- **ROUND**: returns a DATE rounded to the unit specified by the second argument

Date Function Examples:	Result	
SELECT hire_date, ROUND(hire_date, 'Month') FROM employees WHERE department_id = 50;	16-Nov-1999	01-Dec-1999
	17-Oct-1995	01-Nov-1995
	29-Jan-1997	01-Feb-1997

SELECT hire_date, ROUND(hire_date, 'Year') FROM employees WHERE department_id = 50;	16-Nov-1999	01-Jan-2000
	17-Oct-1995	01-Jan-1996
	29-Jan-1997	01-Jan-1997

Date Functions

- **TRUNC:** returns a DATE truncated to the unit specified by the second argument

Date Function Examples:	Result	
SELECT hire_date, TRUNC(hire_date, 'Month') FROM employees WHERE department_id = 50;	16-Nov-1999 17-Oct-1995 29-Jan-1997 ...	01-Nov-1999 01-Oct-1995 01-Jan-1997 ...
SELECT hire_date, TRUNC(hire_date, 'Year') FROM employees WHERE department_id = 50;	16-Nov-1999 17-Oct-1995 29-Jan-1997 ...	01-Jan-1999 01-Jan-1995 01-Jan-1997 ...

If 'month' is the unit specified, the date returned will be the first day of the month for the DATE argument.

If 'Year' is the unit specified, TRUNC will return the first day of the year for the DATE argument.

Date Functions

- Here is an example of a query using multiple date functions
- The output is displayed on the next slide

```
SELECT employee_id, hire_date,  
       ROUND(MONTHS_BETWEEN(SYSDATE, hire_date)) AS TENURE,  
       ADD_MONTHS (hire_date, 6) AS REVIEW,  
       NEXT_DAY(hire_date, 'FRIDAY'), LAST_DAY(hire_date)  
FROM employees  
WHERE MONTHS_BETWEEN (SYSDATE, hire_date) > 36;
```

Date Functions

- The result set from this query returns 20 rows including:

EMPLOYEE_ID	HIRE_DATE	TENURE	REVIEW	NEXT_DAY(HIRE_DATE, 'FRIDAY')	LAST_DAY(HIRE_DATE)
100	17-Jun-1987	348	17-Dec-1987	19-Jun-1987	30-Jun-1987
101	21-Sep-1989	321	21-Mar-1990	22-Sep-1989	30-Sep-1989
102	13-Jan-1993	281	13-Jul-1993	15-Jan-1993	31-Jan-1993
200	17-Sep-1987	345	17-Mar-1988	18-Sep-1987	30-Sep-1987
205	07-Jun-1994	265	07-Dec-1994	10-Jun-1994	30-Jun-1994
206	07-Jun-1994	265	07-Dec-1994	10-Jun-1994	30-Jun-1994
149	29-Jan-2000	197	29-Jul-2000	04-Feb-2000	31-Jan-2000
174	11-May-1996	241	11-Nov-1996	17-May-1996	31-May-1996
176	24-Mar-1998	219	24-Sep-1998	27-Mar-1998	31-Mar-1998
178	24-May-1999	205	24-Nov-1999	28-May-1999	31-May-1999

More than 10 rows available. Increase rows selector to view more rows.

Terminology

Key terms used in this lesson included:

- ADD_MONTHS
- LAST_DAY
- MONTHS_BETWEEN
- NEXT_DAY
- SYSDATE
- ROUND
- TRUNC

Summary

In this lesson, you should have learned how to:

- Select and apply the single-row functions MONTHS_BETWEEN, ADD_MONTHS, NEXT_DAY, LAST_DAY, ROUND, and TRUNC that operate on date data
- Explain how date functions transform Oracle dates into date data or numeric values
- Demonstrate proper use of the arithmetic operators with dates

Summary

In this lesson, you should have learned how to:

- Demonstrate the use of SYSDATE and date functions
- State the implications for world businesses to be able to easily manipulate data stored in date format

The logo for Oracle Academy. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is centered on a light gray background, which is framed by dark gray horizontal bars at the top and bottom.

ORACLE

Academy