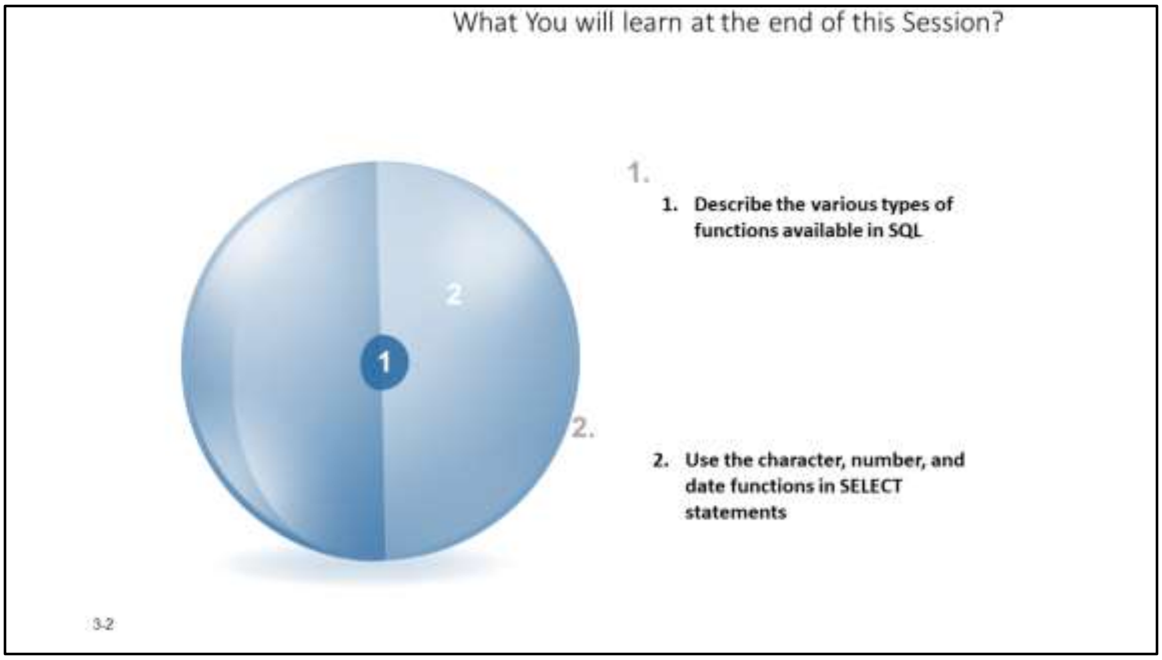
Lesson 2

Using Single-Row Functions to Customize Output



What You will learn at the end of this Session?

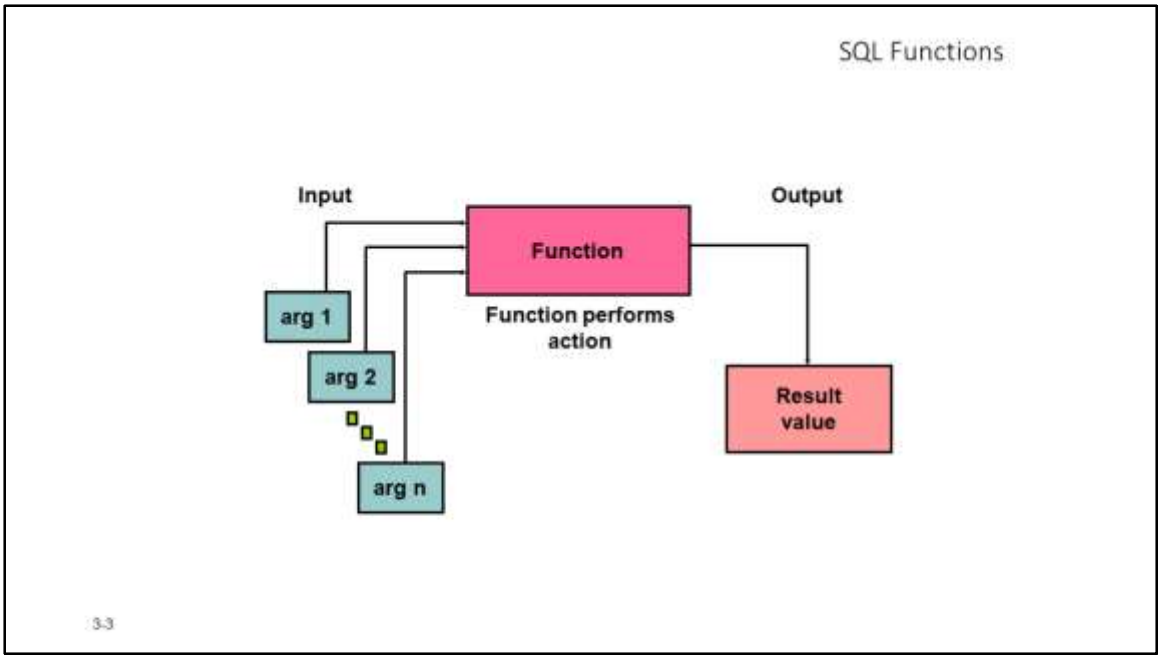
1. Describe the various types of functions available in SQL

2. Use the character, number, and date functions in SELECT statements

3-2

What you will learn at the end of this session?

Functions make the basic query block more powerful, and they are used to manipulate data values. This is the first of two lessons that explore functions. It focuses on single-row character, number, and date functions.


SQL Functions

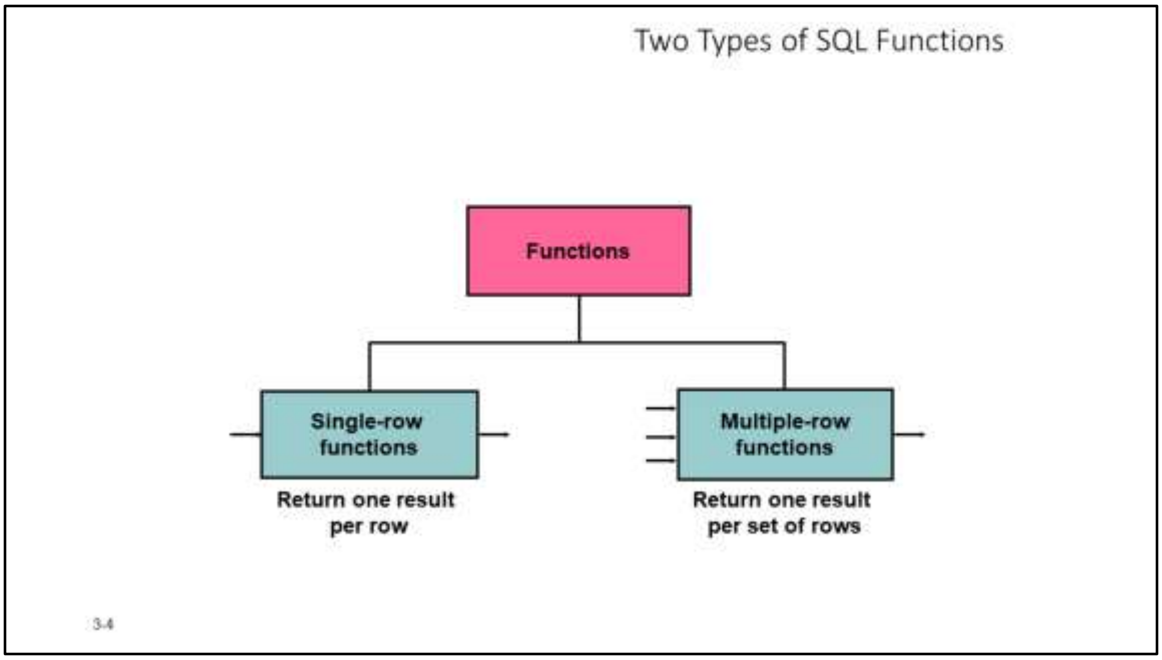
SQL Functions

Functions are a very powerful feature of SQL. They can be used to do the following:

Perform calculations on data
Modify individual data items
Manipulate output for groups of rows
Format dates and numbers for display
Convert column data types

SQL functions sometimes take arguments and always return a value.

Note: If you want to know whether a function is a SQL:2003 compliant function, refer to the "Oracle Compliance to Core SQL:2003" section in *Oracle Database SQL Language Reference* for 10*g* or 11*g* database.


Two Types of SQL Functions

Two Types of SQL Functions

There are two types of functions:

Single-row functions

Multiple-row functions

Single-Row Functions

These functions operate on single rows only and return one result per row. There are different types of single-row functions. This lesson covers the following functions:

Character

Number

Date

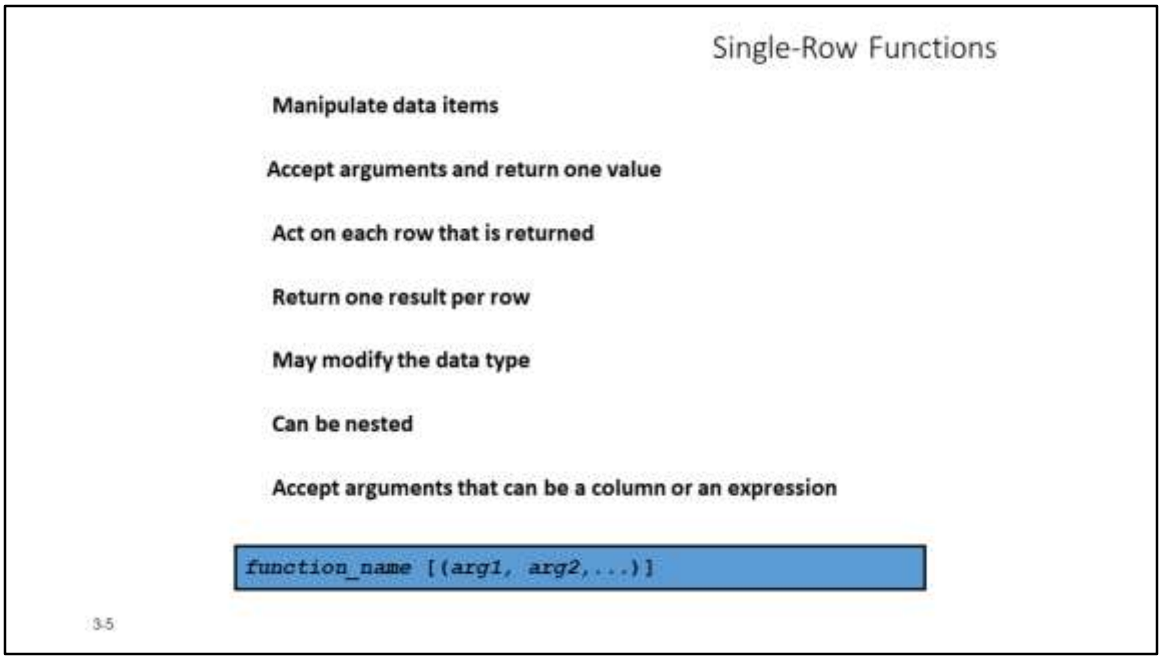
Conversion

General

Multiple-Row Functions

Functions can manipulate groups of rows to give one result per group of rows. These functions are also known as *group functions* (covered in the lesson titled “Reporting Aggregated Data Using the Group Functions”).

Note: For more information and a complete list of available functions and their syntax, see the “Functions” section in *Oracle Database SQL Language Reference* for 10*g* or 11*g* database.


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,...)]
```

3-5

Single-Row Functions

Single-row functions are used to manipulate data items. They accept one or more arguments and return one value for each row that is returned by the query. An argument can be one of the following:

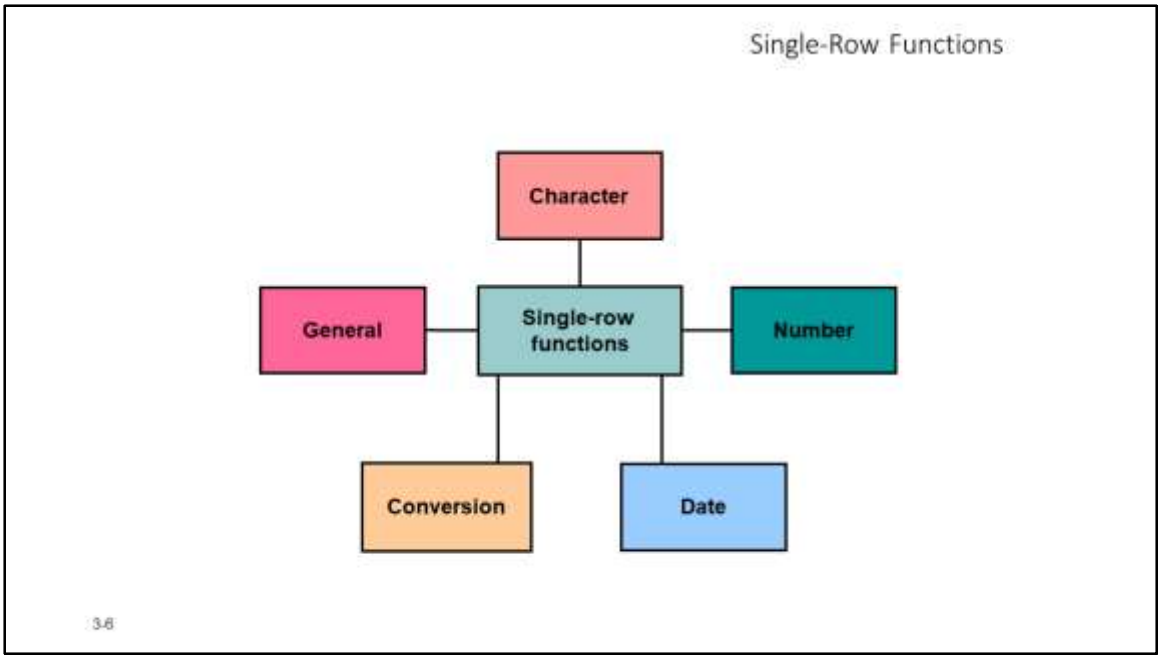
User-supplied constant
Variable value
Column name
Expression

Features of single-row functions include:

Acting on each row that is returned in the query
Returning one result per row
Possibly returning a data value of a different type than the one that is referenced
Possibly expecting one or more arguments
Can be used in `SELECT`, `WHERE`, and `ORDER BY` clauses; can be nested

In the syntax:

`function_name`	Is the name of the function
`arg1, arg2`	Is any argument to be used by the function. This can be represented by a column name or expression.



Single-Row Functions

Single-Row Functions (continued)

This lesson covers the following single-row functions:

Character functions: Accept character input and can return both character and number values

Number functions: Accept numeric input and return numeric values

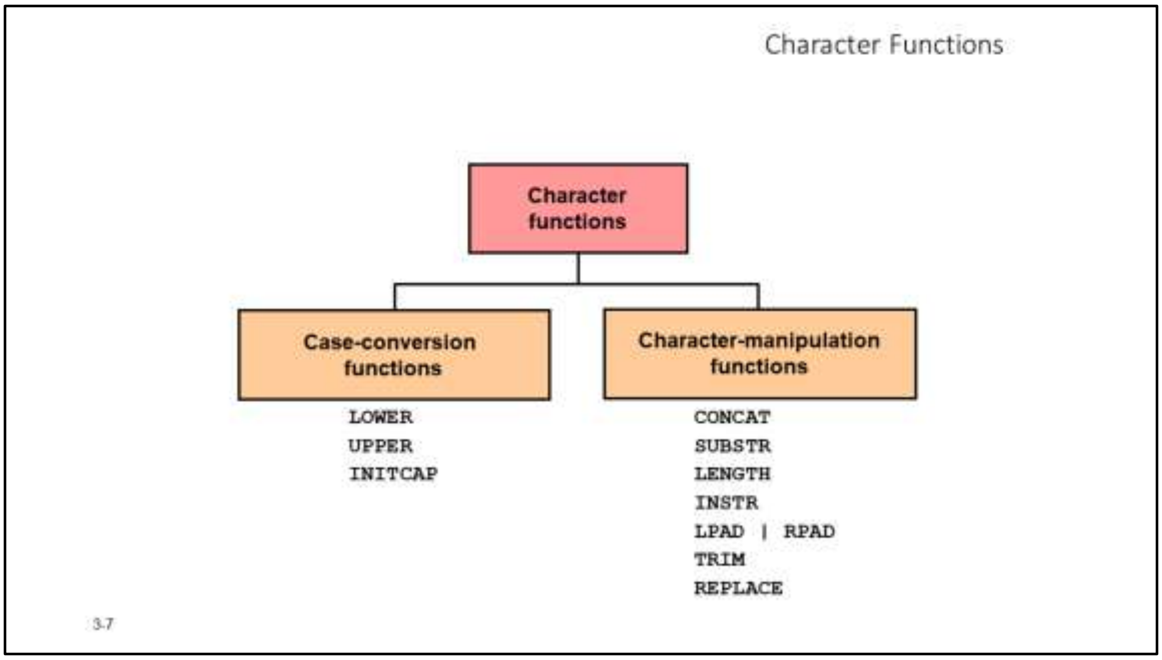
Date functions: Operate on values of the `DATE` data type (All date functions return a value of the `DATE` data type except the `MONTHS_BETWEEN` function, which returns a number.)

The following single-row functions are discussed in the lesson titled “Using Conversion Functions and Conditional Expressions”:

Conversion functions: Convert a value from one data type to another

General functions:

```
NVL
NVL2
NULLIF
COALESCE
CASE
DECODE
```



Character Functions

Single-row character functions accept character data as input and can return both character and numeric values. Character functions can be divided into the following:

Case-conversion functions

Character-manipulation functions

Function	Purpose
`LOWER(`*`column`*`|`*`expression`*`)`	Converts alpha character values to lowercase
`UPPER(`*`column`*`|`*`expression`*`)`	Converts alpha character values to uppercase
`INITCAP(`*`column`*`|`*`expression`*`)`	Converts alpha character values to uppercase for the first letter of each word; all other letters in lowercase
`CONCAT(`*`column1`*`|`*`expression1`*`, `*`column2`*`|`*`expression2`*`)`	Concatenates the first character value to the second character value; equivalent to concatenation operator (||)
`SUBSTR(`*`column`*`|`*`expression`*`,`*`m`*`[,`*`n`*`])`	Returns specified characters from character value starting at character position *m*, *n* characters long (If *m* is negative, the count starts from the end of the character value. If *n* is omitted, all characters to the end of the string are returned.)

Note: The functions discussed in this lesson are only some of the available functions.

Character Functions (continued)

Function	**Purpose**
`LENGTH(column|expression)`	Returns the number of characters in the expression
`INSTR(column|expression, ’string’, [,m], [n] )`	Returns the numeric position of a named string. Optionally, you can provide a position *m* to start searching, and the occurrence *n* of the string. *m* and *n* default to 1, meaning start the search at the beginning of the string and report the first occurrence.

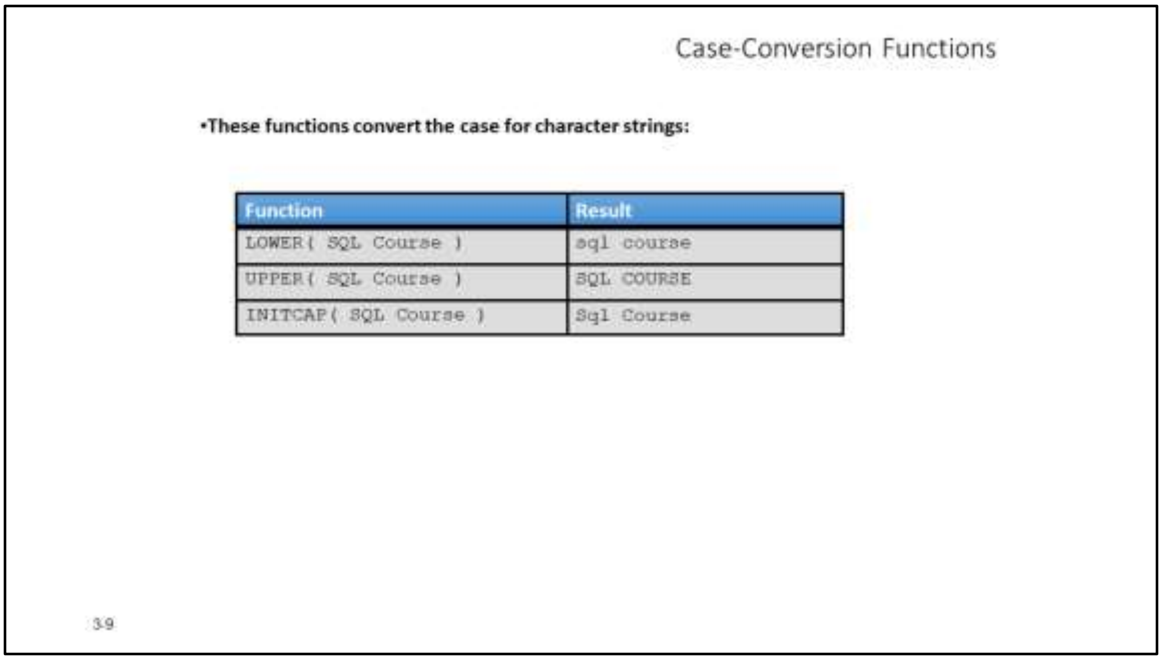
`LPAD(column|expression, n, 'string')`
`RPAD(column|expression, n, 'string')`
	Returns an expression left-padded to length of *n* characters with a character expression.
	Returns an expression right-padded to length of *n* characters with a character expression.

`TRIM(leading|trailing|both, trim_character FROM trim_source)`	Enables you to trim leading or trailing characters (or both) from a character string. If *trim_character* or *trim_source* is a character literal, you must enclose it in single quotation marks. This is a feature that is available in Oracle8*i* and later versions.
`REPLACE(text, search_string, replacement_string)`	Searches a text expression for a character string and, if found, replaces it with a specified replacement string

Note: Some of the functions that are fully or partially SQL:2003 compliant are:

```
UPPER
LOWER
TRIM
LENGTH
SUBSTR
INSTR
```

For more information, refer to the “Oracle Compliance to Core SQL:2003” section in *Oracle Database SQL Language Reference* for 10*g* or 11*g* database.


Case-Conversion Functions

•These functions convert the case for character strings:

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

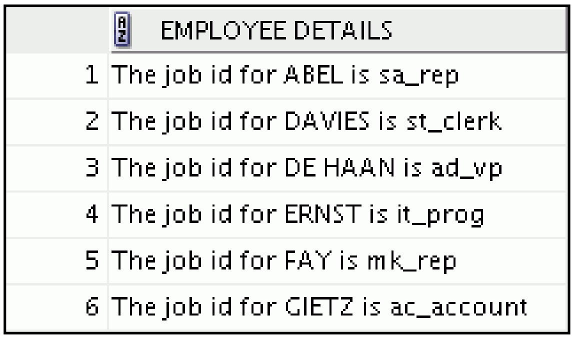
3-9

Case-Conversion Functions

LOWER, UPPER, and INITCAP are the three case-conversion functions.

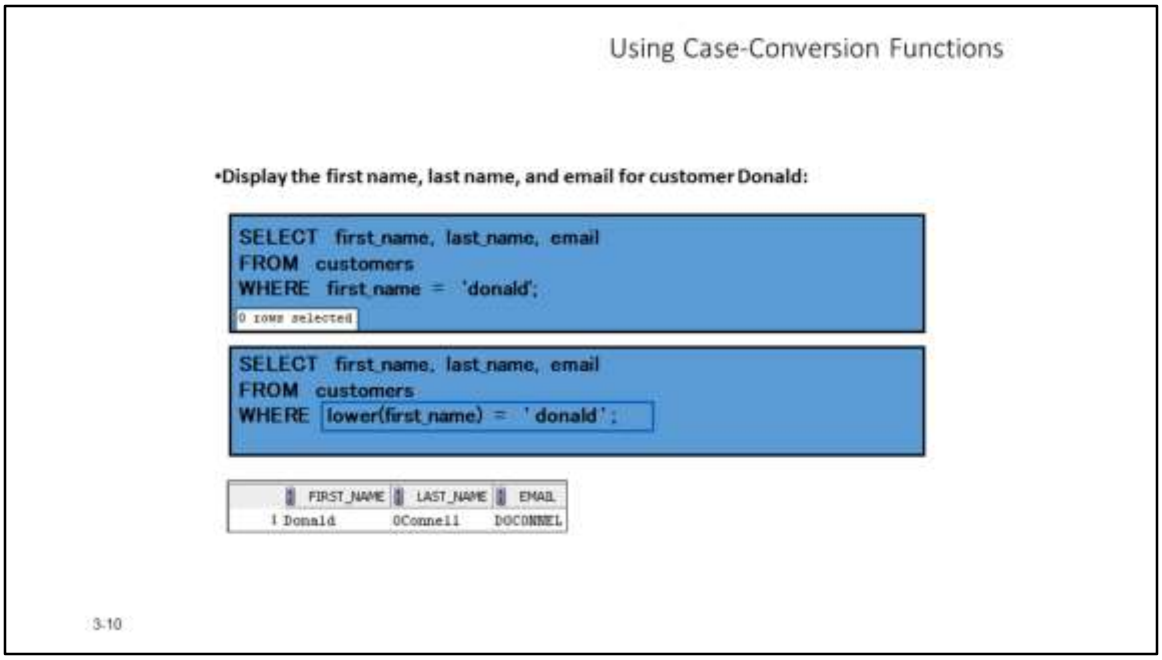
- LOWER: Converts mixed-case or uppercase character strings to lowercase
- UPPER: Converts mixed-case or lowercase character strings to uppercase
- INITCAP: Converts the first letter of each word to uppercase and the remaining letters to lowercase

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



	EMPLOYEE DETAILS
1	The job id for ABEL is sa_rep
2	The job id for DAVIES is st_clerk
3	The job id for DE HAAN is ad_vp
4	The job id for ERNST is it_prog
5	The job id for FAY is mk_rep
6	The job id for GIETZ is ac_account

…



Using Case-Conversion Functions

•Display the first name, last name, and email for customer Donald:

```
SELECT  first_name,  last_name,  email
FROM  customers
WHERE  first_name  =  'donald';
```

```
0 rows selected
```

```
SELECT  first_name,  last_name,  email
FROM  customers
WHERE  lower(first_name)  =  'donald';
```

	FIRST_NAME	LAST_NAME	EMAIL
1	Donald	OConnell	DOCONNEL

3-10

Using Case-Conversion Functions

The slide example displays the first name, last name, and email for customer Donald:

The WHERE clause of the first SQL statement specifies the employee name as donald. Because all the data in the CUSTOMER table is stored in proper case, the name donald does not find a match in the table, and no rows are selected.

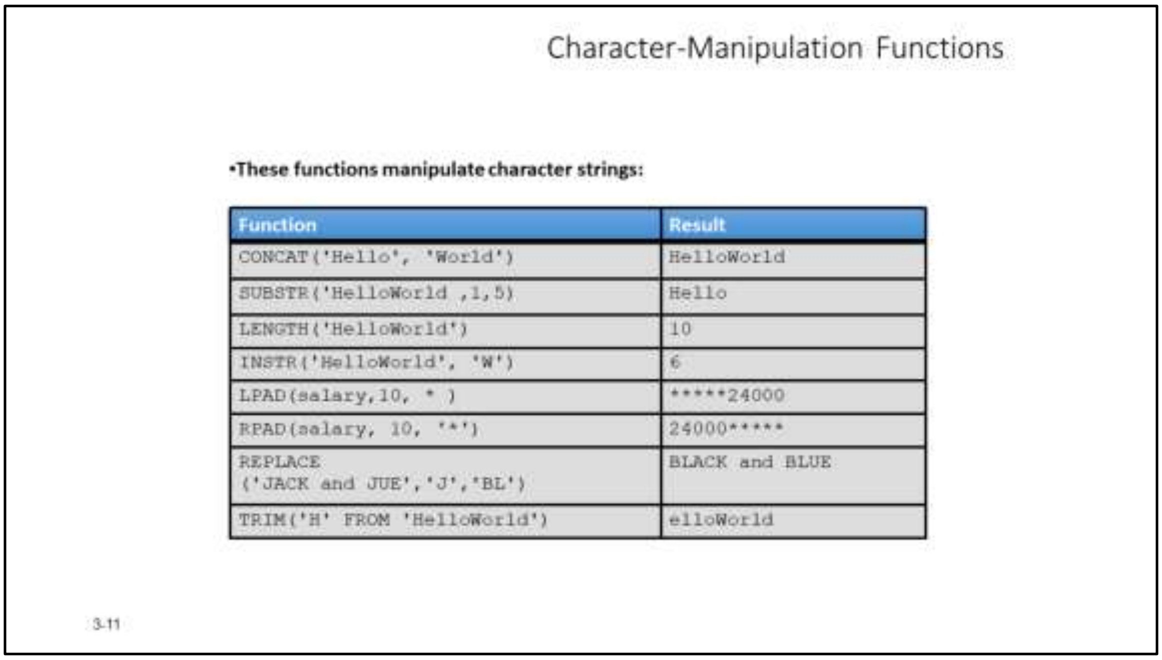
The WHERE clause of the second SQL statement specifies that the customer name in the CUSTOMERS table is compared to donald, converting the FIRST_NAME column to lowercase for comparison purposes. Because both names are now lowercase, a match is found and one row is selected. The WHERE clause can be rewritten in the following manner to produce the same result:

```
...WHERE first_name = ‘Donald'
```

The name in the output appears as it was stored in the database.

To display a name in uppercase, use the UPPER function in the SELECT statement.

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



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`

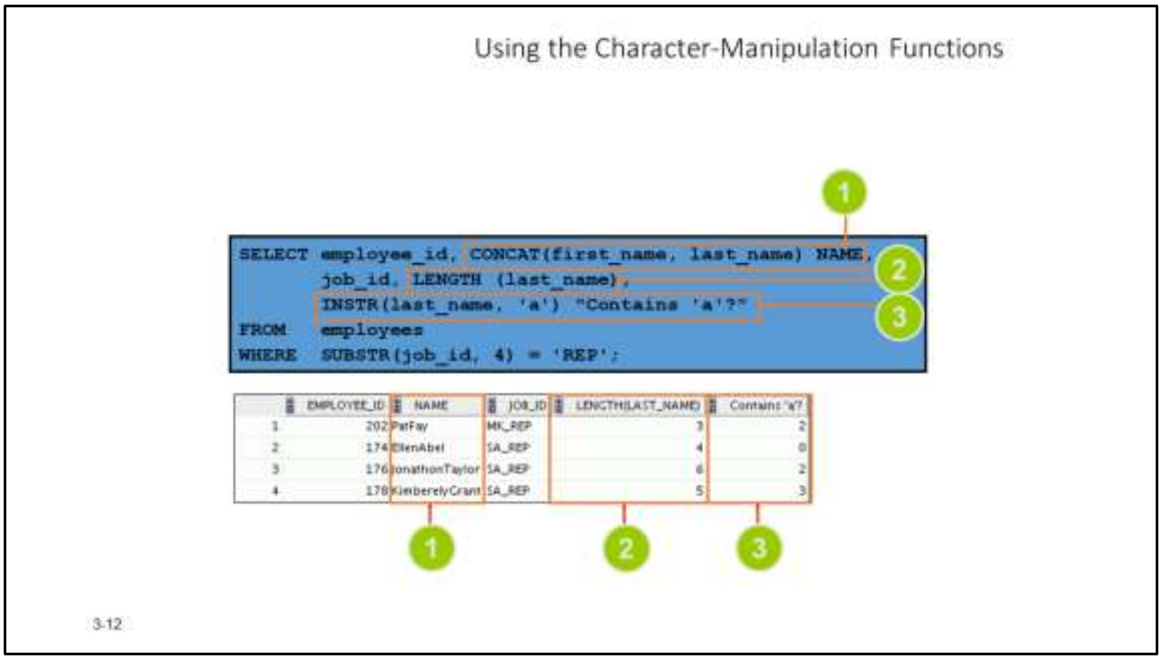
3-11

Character-Manipulation Functions

`CONCAT`, `SUBSTR`, `LENGTH`, `INSTR`, `LPAD`, `RPAD`, and `TRIM` are the character-manipulation functions that are covered in this lesson.

- `CONCAT`: Joins values together (You are limited to using two parameters with `CONCAT`.)
- `SUBSTR`: Extracts a string of determined length
- `LENGTH`: Shows the length of a string as a numeric value
- `INSTR`: Finds the numeric position of a named character
- `LPAD`: Returns an expression left-padded to the length of *n* characters with a character expression
- `RPAD`: Returns an expression right-padded to the length of *n* characters with a character expression
- `TRIM`: Trims leading or trailing characters (or both) from a character string (If *`trim_character`* or *`trim_source`* is a character literal, you must enclose it within single quotation marks.)

Note: You can use functions such as `UPPER` and `LOWER` with ampersand substitution. For example, use `UPPER('&job_title')` so that the user does not have to enter the job title in a specific case.



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';
```

	EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
1	202	PatFay	MK_REP	3	2
2	174	EllenAbel	SA_REP	4	0
3	176	JonathonTaylor	SA_REP	6	2
4	178	KimberelyGrant	SA_REP	5	3

3-12

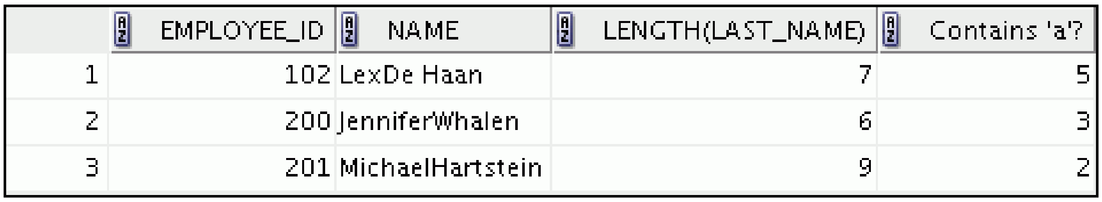
Using the Character-Manipulation Functions

The example in the slide displays employee first names and last names joined together, the length of the employee last name, and the numeric position of the letter "a" in the employee last name for all employees who have the string, `REP`, contained in the job ID starting at the fourth position of the job ID.

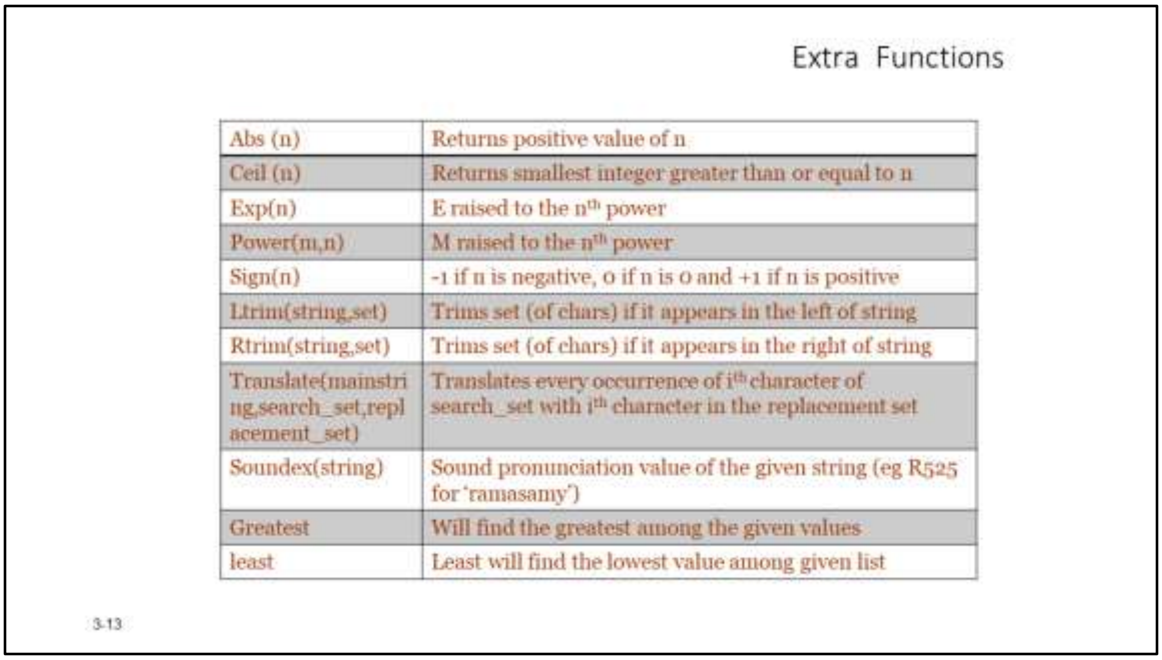
Example:

Modify the SQL statement in the slide to display the data for those employees whose last names end with the letter "n."

```
SELECT employee_id, CONCAT(first_name, last_name) NAME,
LENGTH (last_name), INSTR(last_name, 'a') "Contains 'a'?"
FROM   employees
WHERE  SUBSTR(last_name, -1, 1) = 'n';
```



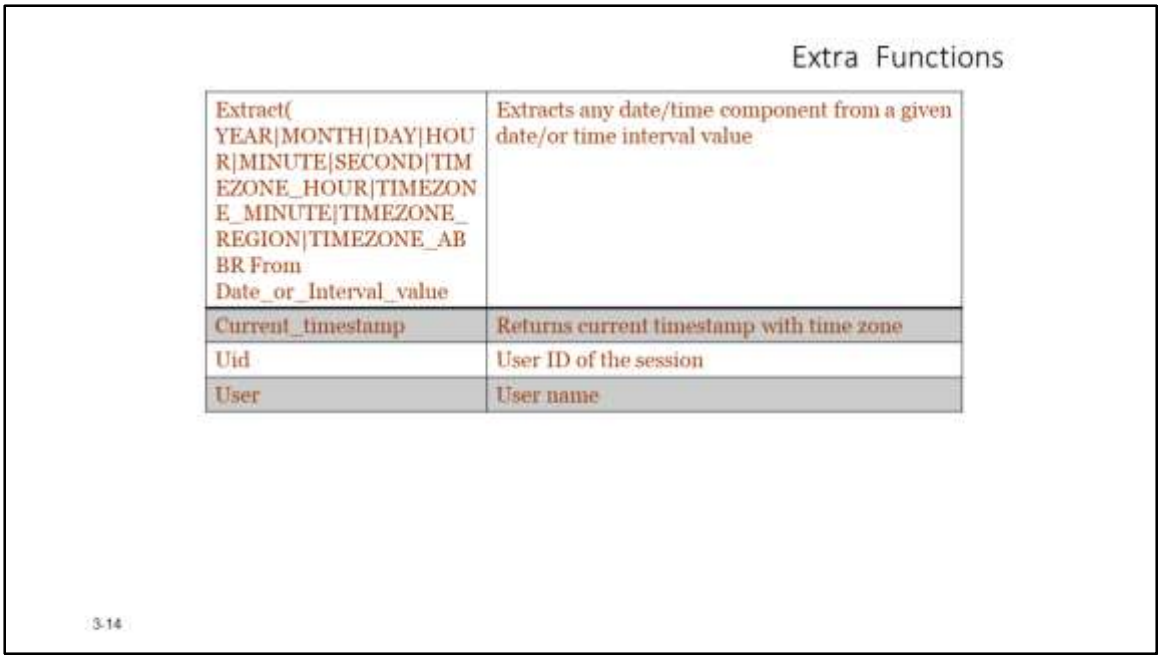
	EMPLOYEE_ID	NAME	LENGTH(LAST_NAME)	Contains 'a'?
1	102	LexDe Haan	7	5
2	200	JenniferWhalen	6	3
3	201	MichaelHartstein	9	2



Extra Functions

Abs (n)	Returns positive value of n
Ceil (n)	Returns smallest integer greater than or equal to n
Exp(n)	E raised to the nth power
Power(m,n)	M raised to the nth power
Sign(n)	-1 if n is negative, 0 if n is 0 and +1 if n is positive
Ltrim(string,set)	Trims set (of chars) if it appears in the left of string
Rtrim(string,set)	Trims set (of chars) if it appears in the right of string
Translate(mainstring,search_set,replacement_set)	Translates every occurrence of ith character of search_set with ith character in the replacement set
Soundex(string)	Sound pronunciation value of the given string (eg R525 for 'ramasamy')
Greatest	Will find the greatest among the given values
least	Least will find the lowest value among given list

3-13



Extra Functions

Extract(YEAR|MONTH|DAY|HOUR|MINUTE|SECOND|TIMEZONE_HOUR|TIMEZONE_MINUTE|TIMEZONE_REGION|TIMEZONE_ABBR From Date_or_Interval_value	Extracts any date/time component from a given date/or time interval value
Current_timestamp	Returns current timestamp with time zone
Uid	User ID of the session
User	User name

3-14