Database Programming

Case and Character Manipulation

Objectives

This lesson covers the following objectives:

- Select and apply single-row functions that perform case conversion and/or character manipulation
- Select and apply character case-manipulation functions LOWER, UPPER, and INITCAP in a SQL query
- Select and apply character-manipulation functions CONCAT, SUBSTR, LENGTH, INSTR, LPAD, RPAD, TRIM, and REPLACE in a SQL query
- Write flexible queries using substitution variables

Purpose

Have you ever thought about the different ways in which we present ourselves? We dress up for special occasions, dress casually for play, and put on uniforms for sports events and band concerts. Being able to change the way we look for different situations is important. How would you choose to present yourself for a job interview?

Purpose (cont.)

Being able to change the way in which data is presented is important when dealing with data from a database. Most of the time in SQL, we need to change the way that data appears depending on the requirements of the task we are trying to accomplish.

In this lesson, you will learn several ways in which to transform data to fit a particular situation.

DUAL Table

The DUAL table has one row called "X" and one column called "DUMMY." The DUAL table is used to create SELECT statements and execute functions not directly related to a specific database table. Queries using the DUAL table return one row as a result. DUAL can be useful to do calculations and also to evaluate expressions that are not derived from a table.

DUMMY
X

DUAL Table (cont.)

DUAL will be used to learn many of the single-row functions. In this example the DUAL table is used to execute a SELECT statement that contains a calculation. As you can see the SELECT statement returns a value that does not exist in the DUAL table. The value returned is a result of the calculation executed.

```
SELECT (319/29) + 12
FROM DUAL;
```

(319/29)+12
23

Single-Row Character Functions

Single-row character functions are divided into two categories:

- Functions that convert the case of character strings
- Functions that can join, extract, show, find, pad, and trim character strings

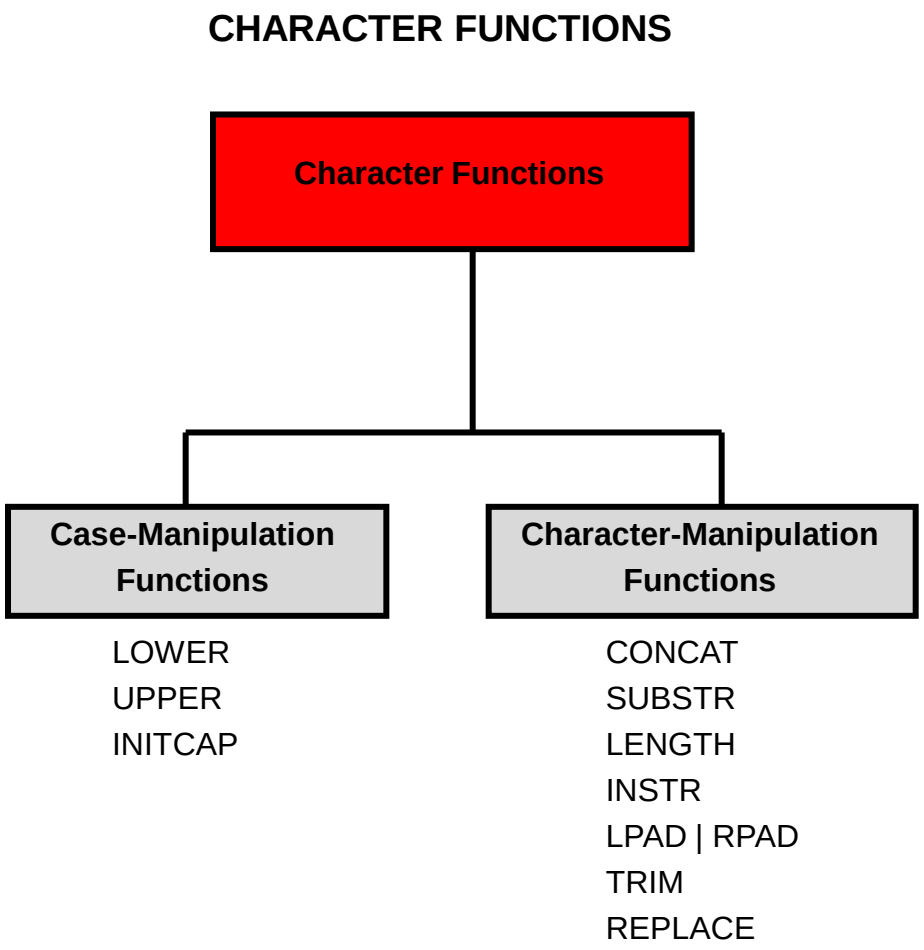
Single-row functions can be used in the `SELECT`, `WHERE`, and `ORDER` BY clauses.

Single-Row Character Functions (cont.)

Case-manipulation functions are important because you may not always know in which case (upper, lower, or mixed) the data is stored in the database. Case manipulation allows you to temporarily convert the database data to a case of your choosing. Mismatches between database case storage and query case requests are avoided.

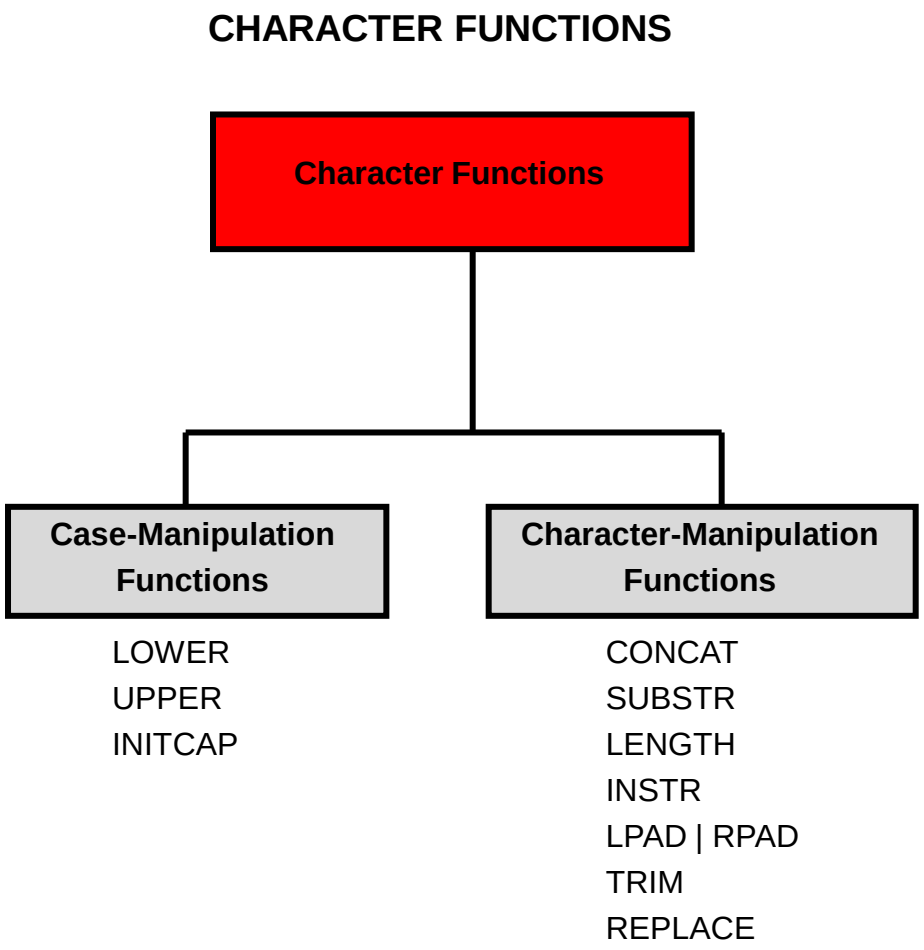
Case Manipulation Functions

Case-manipulation functions are used to convert from lower to upper or mixed case. These conversions can be used to format the output and can also be used to search for specific strings.



Case Manipulation Functions (cont.)

Case-manipulation functions can be used in most parts of a SQL statement.



Case Manipulation Functions (cont.)

Case-manipulation functions are often helpful when you are searching for data and you do not know whether the data you are looking for is in upper or lower case. From the point of view of the database, ‘V’ and ‘v’ are NOT the same character and, as such, you need to search using the correct case.

`LOWER(column | expression)` converts alpha characters to lower-case.

```
SELECT title
FROM d_cds
WHERE LOWER(title) = 'carpe diem';
```

Case Manipulation Functions (cont.)

`UPPER(column | expression)` converts alpha characters to upper-case.

```
SELECT title
FROM d_cds
WHERE UPPER(title) = 'CARPE DIEM';
```

`INITCAP(column | expression)` converts alpha character values to uppercase for the first letter of each word.

```
SELECT title
FROM d_cds
WHERE INITCAP(title) = 'Carpe Diem';
```

Character Manipulation Functions

Character-manipulation functions are used to extract, change, format, or alter in some way a character string.

One or more characters or words are passed into the function and the function will then perform its functionality on the input character strings and return the changed, extracted, counted, or altered value.

Character Manipulation Functions (cont.)

CONCAT: Joins two values together.

SUBSTR: Extracts a string of a determined length.

Function	Result
CONCAT('Hello', 'World')	HelloWorld
SUBSTR('HelloWorld', 1, 5)	Hello

Character Manipulation Functions (cont.)

- LENGTH: Shows the length of a string as a number value.
- INSTR: Finds the numeric position of a named character.
- LPAD: Pads the left side of a character, resulting in a right-justified value.

Function	Result
LENGTH('HelloWorld')	10
INSTR('HelloWorld','W')	6
LPAD(salary, 10,'*')	*****24000

Character Manipulation Functions (cont.)

- RPAD: Pads the right-hand side of a character, resulting in a left-justified value.
- TRIM: Removes all specified characters from either the beginning or the ending of a string. The syntax for the trim function is:

```
TRIM( [ leading | trailing | both
[character(s) to be removed ] ] string to trim
```

Function	Result
RPAD(salary, 10, '*')	24000*****
TRIM('H' FROM 'HelloWorld')	elloWorld

Character Manipulation Functions (cont.)

REPLACE: Replaces a sequence of characters in a string with another set of characters. The syntax for the REPLACE function is:

```
REPLACE (string1, string_to_replace, [replacement_string] )
```

- string1 is the string that will have characters replaced in it;
- string_to_replace is the string that will be searched for and taken out of string1;
- [replacement_string] is the new string to be inserted in string1.

Character Manipulation Functions (cont.)

```
SELECT REPLACE('JACK and JUE','J','BL') "Changes"
FROM DUAL;
```

Using Column Aliases With Functions

All functions operate on values that are in parentheses, and each function name denotes its purpose, which is helpful to remember when constructing a query.

Often times a column alias is used to name a function. When a column alias is used, the column alias appears in the output instead of the actual function syntax.

Using Column Aliases With Functions (cont.)

In the following examples, the alias "User Name" has replaced the function syntax in the first query.

By default, the column name in a SELECT statement appears as the column heading. In the second query example, however, there is no column in the table for the results produced, so the query syntax is used instead.

Using Column Aliases With Functions (continued)

```
SELECT LOWER (last_name)||LOWER(SUBSTR(first_name,1,1))
AS "User Name"
FROM f_staffs;
```

User Name
does
millerb
tuttlem

```
SELECT LOWER (last_name)||LOWER(SUBSTR(first_name,1,1))
FROM f_staffs;
```

LOWER (last_name)||LOWER(SUBSTR(first_name,1,1))
does
millerb
tuttlem

Substitution Variables

Occasionally you may need to run the same query with many different values to get different result sets. Imagine for instance if you had to write a report of employees and their departments, but the query must only return data for one department at a time. Without the use of substitution variables, this request would mean you would have to repeatedly edit the same statement to change the WHERE-clause.

Substitution Variables (cont.)

Luckily for us, Oracle Application Express supports substitution variables. To use them, all you have to do is replace the hardcoded value in your statement with a **:named_variable**. Oracle Application Express will then ask you for a value when you execute your statement.

Substitution Variables (cont.)

So this original query:

```
SELECT first_name, last_name, salary, department_id
FROM    employees
WHERE department_id = 10 (and then 20, 30, 40…)
```

could be re-written to:

```
SELECT first_name, last_name, salary, department_id
FROM    employees
WHERE department_id = :dept_id
```

Note the use of **:** in front of dept_id. It is the colon that is the magic bit and makes Oracle Application Express recognize the text that follows as a variable.

Substitution Variables (cont.)

Substitution variables are treated as character strings in Oracle Application Express, which means that when passing in character or date values, you do not need the single quotation marks that you would normally use to enclose the strings.

So a WHERE-clause would look like this:

```
SELECT *
FROM    employees
WHERE last_name = :l_name
```

Substitution Variables (cont.)

When you click Run, a pop-up like the following is displayed by Oracle Application Express:

Terminology

Key terms used in this lesson included:

- Character functions
- CONCAT
- DUAL
- Expression
- Format
- INITCAP
- Input
- INSTR

Terminology (cont.)

Key terms used in this lesson included:

- LENGTH
- LOWER
- LPAD
- Output
- REPLACE
- RPAD
- Single-row functions
- SUBSTR

Terminology

Key terms used in this lesson included:

- TRIM
- UPPER
- Substitution variable

Summary

In this lesson, you should have learned how to:

- Select and apply single-row functions that perform case conversion and/or character manipulation
- Select and apply character case-manipulation functions LOWER, UPPER, and INITCAP in a SQL query
- Select and apply character-manipulation functions CONCAT, SUBSTR, LENGTH, INSTR, LPAD, RPAD, TRIM, and REPLACE in a SQL query
- Write flexible queries using substitution variables