This section will focus on string functions that expand our way of selection data and work with text and strings

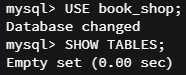
Section slides: <http://webdev.slides.com/coltsteele/mysql-99-101>

Reminder: how to source SQL files in goorm IDE: <https://www.youtube.com/watch?v=DjhthBTXvXg>

Loading Our Book Data

* We’ll be working with books data, which are available in the books\_data.sql file
  + In the code provided, we get book IDs (primary key), titles, author first and last names, year released, quantity in stock, and number of pages
  + In this case, the years will be INT instead of DATE datatypes since we haven’t learned to work with dates yet
* We’ll also be creating a new database called book\_shop





* The books table is a bit messy, and this is deliberate. Going forward we’ll learn how to work with messy text data to do really neat stuff in MySQL
* Code summary

DROP DATABASE IF EXISTS book\_shop;

CREATE DATABASE book\_shop;

USE book\_shop;

CREATE TABLE books

(

book\_id INT NOT NULL AUTO\_INCREMENT,

title VARCHAR(100),

author\_fname VARCHAR(100),

author\_lname VARCHAR(100),

released\_year INT,

stock\_quantity INT,

pages INT,

PRIMARY KEY(book\_id)

);

INSERT INTO books (title, author\_fname, author\_lname, released\_year, stock\_quantity, pages)

VALUES

('The Namesake', 'Jhumpa', 'Lahiri', 2003, 32, 291),

('Norse Mythology', 'Neil', 'Gaiman',2016, 43, 304),

('American Gods', 'Neil', 'Gaiman', 2001, 12, 465),

('Interpreter of Maladies', 'Jhumpa', 'Lahiri', 1996, 97, 198),

('A Hologram for the King: A Novel', 'Dave', 'Eggers', 2012, 154, 352),

('The Circle', 'Dave', 'Eggers', 2013, 26, 504),

('The Amazing Adventures of Kavalier & Clay', 'Michael', 'Chabon', 2000, 68, 634),

('Just Kids', 'Patti', 'Smith', 2010, 55, 304),

('A Heartbreaking Work of Staggering Genius', 'Dave', 'Eggers', 2001, 104, 437),

('Coraline', 'Neil', 'Gaiman', 2003, 100, 208),

('What We Talk About When We Talk About Love: Stories', 'Raymond', 'Carver', 1981, 23, 176),

("Where I'm Calling From: Selected Stories", 'Raymond', 'Carver', 1989, 12, 526),

('White Noise', 'Don', 'DeLillo', 1985, 49, 320),

('Cannery Row', 'John', 'Steinbeck', 1945, 95, 181),

('Oblivion: Stories', 'David', 'Foster Wallace', 2004, 172, 329),

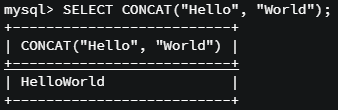
('Consider the Lobster', 'David', 'Foster Wallace', 2005, 92, 343);

Working with CONCAT

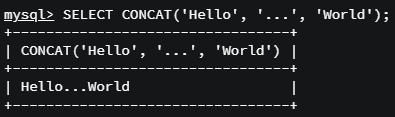
* Documentation: <https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_concat>
* The **CONCAT()** function combines strings together
  + For example, in our table we have author first name and last names separate. This can be useful in certain applications
  + Using the CONCAT function, we can concatenate the first and last names to get a full name
* Let’s start by simply doing a compound selection of the first and last names



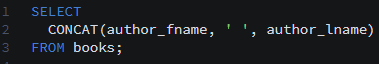
* The syntax for the combination is CONCAT(column, anotherColumn, yetAnotherColumn)
  + However, this will just slam the columns together with no space or separators. Sometimes you want this, but oftentimes you do not
* For this, you can do something like the following:
  + CONCAT(column, 'text', anotherColumn, 'more text')
  + So you add a space, you can do this:
    - CONCAT(author\_fname, ' ', author\_lname)
* How do we actually write this in MySQL. The pseudocode above us NOT valid syntax (for instance, we did not tell SQL which table to select from). We cannot just run CONCAT() on its own – rather, it needs to be run as a SELECT
* Let’s see a simple example of SELECT CONCAT() with plain text. Notice how it smashes them together.

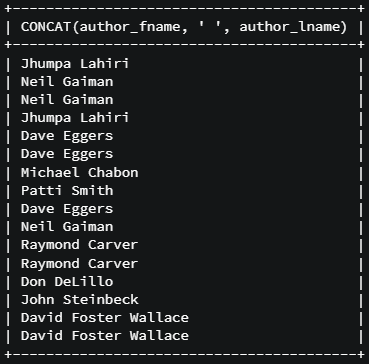


* Now let’s add some characters in between



* Excellent, we’re ready to go back to our table and combine the “author\_fname” and “author\_lname” columns using CONCAT, adding in a space between them
  + Note that the syntax for the selection needs to include the CONCAT() command as well as the table that is being selected from



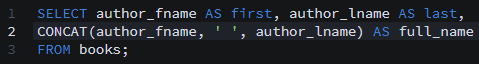


* + Important note: This DOES NOT modify the data in the actual table
* Also notice how ugly the concatenated title looks. We can rename it using AS:





* + Much better!
* String functions can be executed in parallel within the same SELECT statement. In this example, we will select both the “author\_fname” and “author\_lname” columns and rename them to “first” and “last” respectively, and then we will concatenate them with a space to create a full name that we will also call “full\_name”.
  + We could also have multiple CONCAT() statements within each SELECT clause





* The CONCAT\_WS() function stands for *concatenate with separator*. If you want to concatenate multiple items together with a consistent character or space between them, you can use this function as a shortcut.
  + Suppose we wanted to list the title, author first name, and author last name separated by dashes. The “classic” way of doing this involves quite a bit of typing



* + Using **CONCAT\_WS(*separator*, str1, str2)** this is a bit easier





* Code summary for CONCAT() and CONCAT\_WS()

SELECT author\_fname, author\_lname FROM books;

CONCAT(x,y,z) // from slides

CONCAT(column, anotherColumn) // from slides

CONCAT(author\_fname, author\_lname)

CONCAT(column, 'text', anotherColumn, 'more text')

CONCAT(author\_fname, ' ', author\_lname)

CONCAT(author\_fname, author\_lname); // invalid syntax

SELECT CONCAT('Hello', 'World');

SELECT CONCAT('Hello', '...', 'World');

SELECT

CONCAT(author\_fname, ' ', author\_lname)

FROM books;

SELECT

CONCAT(author\_fname, ' ', author\_lname)

AS 'full name'

FROM books;

SELECT author\_fname AS first, author\_lname AS last,

CONCAT(author\_fname, ' ', author\_lname) AS full

FROM books;

SELECT author\_fname AS first, author\_lname AS last,

CONCAT(author\_fname, ', ', author\_lname) AS full

FROM books;

SELECT CONCAT(title, '-', author\_fname, '-', author\_lname) FROM books;

SELECT

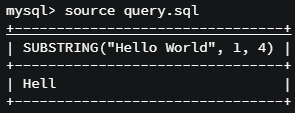
CONCAT\_WS(' - ', title, author\_fname, author\_lname)

FROM books;

Introduction SUBSTRING()

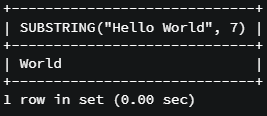
* Documentation: <https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_substring>
* SUBSTRING allows you to work with individual parts of a string
  + Indexing in MySQL starts at 1, not at 0 like other programming languages
* Syntax options (depends on intention) -
  + [SUBSTRING(***str***,***pos***)](https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_substring)
    - Selection will start at the *pos* position and continue to the end
  + [SUBSTRING(***str*** FROM ***pos***)](https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_substring)
  + [SUBSTRING(***str***,***pos***,***len***)](https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_substring)
    - Selection is inclusive of both the *pos* and *len* positions
  + [SUBSTRING(***str*** FROM ***pos*** FOR ***len***)](https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_substring)
* Trivial example – selecting a slice of the string:





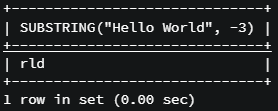
* Another example – selecting a character and all characters onward:





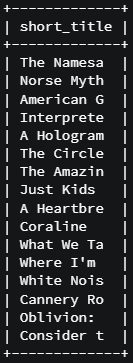
* Another example – using negative indices
  + In this case, the selection starts from the end of the string, counts *back* to the negative index provided, then selects forward to the end





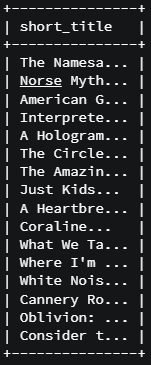
* Let’s go back to our books table and work with that data. We can snag a substring from each book in our list as follows:





* **Any string operators can be combined**! Suppose we wanted to add ‘…’ after our substring short title. Let’s try that.
  + The approach is to wrap our SUBSTRING() function call within the CONCAT() function call. This approach is similar when combining any string functions





* Code summary:

SELECT SUBSTRING('Hello World', 1, 4);

SELECT SUBSTRING('Hello World', 7);

SELECT SUBSTRING('Hello World', 3, 8);

SELECT SUBSTRING('Hello World', 3);

SELECT SUBSTRING('Hello World', -3);

SELECT SUBSTRING('Hello World', -7);

SELECT title FROM books;

SELECT SUBSTRING("Where I'm Calling From: Selected Stories", 1, 10);

SELECT SUBSTRING(title, 1, 10) FROM books;

SELECT SUBSTRING(title, 1, 10) AS 'short title' FROM books;

SELECT SUBSTR(title, 1, 10) AS 'short title' FROM books;

SELECT CONCAT

    (

        SUBSTRING(title, 1, 10),

        '...'

    )

FROM books;

source book\_code.sql

SELECT CONCAT

    (

        SUBSTRING(title, 1, 10),

        '...'

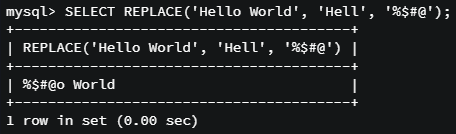
    ) AS 'short title'

FROM books;

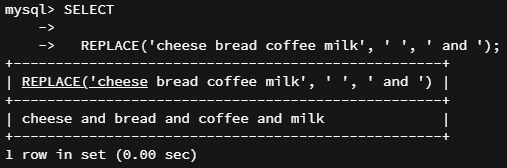
source book\_code.sql

Introducing REPLACE()

* The REPLACE function replaces parts of a string
  + Allows you to do things like replace spaces with commas
* Documentation: <https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_replace>
* Syntax: SELECT REPLACE(“full\_string\_to\_search”, substring\_to\_replace, 'replacement\_string');
  + Example:

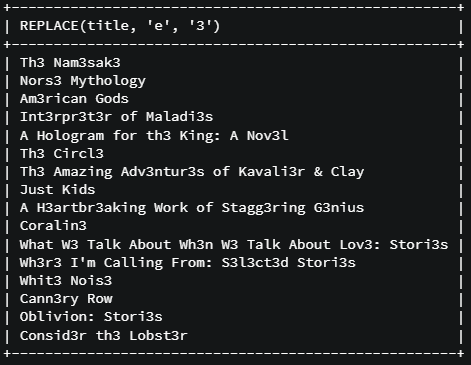


* As with other string functions, REPLACE can be used to replace multiple things in one shot. In this example, all blank spaces are replaced by ‘ and ‘



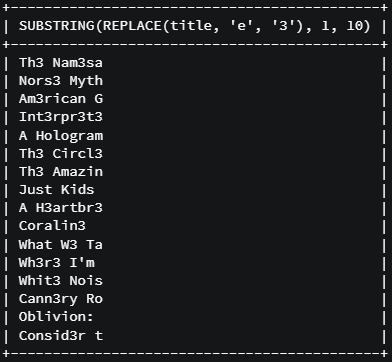
* Going back to our book data, let’s replace all “e’s” with 3’s





* We can combine REPLACE with other functions. Let’s try using REPLACE and SUBSTRING. Here, we will replace all “e’s” with “3’s” in the book titles, and then select a substring of the replaced titles from positions 1 to 10





* Code summary

SELECT REPLACE('Hello World', 'Hell', '%$#@');

SELECT REPLACE('Hello World', 'l', '7');

SELECT REPLACE('Hello World', 'o', '0');

SELECT REPLACE('HellO World', 'o', '\*');

SELECT

REPLACE('cheese bread coffee milk', ' ', ' and ');

SELECT REPLACE(title, 'e ', '3') FROM books;

-- SELECT

-- CONCAT

-- (

-- SUBSTRING(title, 1, 10),

-- '...'

-- ) AS 'short title'

-- FROM books;

SELECT

SUBSTRING(REPLACE(title, 'e', '3'), 1, 10)

FROM books;

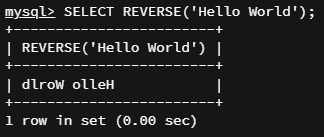
SELECT

SUBSTRING(REPLACE(title, 'e', '3'), 1, 10) AS 'weird string'

FROM books;

The REVERSE Function

* The REVERSE function simply reverses strings
  + Casing and spaces are preserved
* Documentation: <https://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_reverse>
* Simple example:

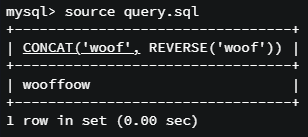


* Example: Let’s reverse every author’s first name in our table





* Another example: making a palindrome using CONCAT and REVERSE



* Kicking it up a notch: palindromes of author’s names





* Code summary

SELECT REVERSE('Hello World');

SELECT REVERSE('meow meow');

SELECT REVERSE(author\_fname) FROM books;

SELECT CONCAT('woof', REVERSE('woof'));

SELECT CONCAT(author\_fname, REVERSE(author\_fname)) FROM books;