Spring 2022 courses

IN PROGRESS db quiz 5

Settings

- A question has only ONE right answer unless otherwise noted
- This quiz covers the last few DataCamp assignments (Introduction to SQL) and SQLite basics
- After the first play, the quiz will be opened for unlimited play
- Let me know if you have any comments or corrections

SQL commands cannot go over multiple lines (no linebreaks allowed)

FALSE

Feedback: when you break the line, SQL (or SQLite) follows you (with a different prompt, on the console) until you close the command with a semi-colon (;).

SQLite commands always begin with a dot (.)

FALSE

Feedback: E.g. .database, .tables, .show, and many other commands on the shell. When you are NOT on the shell, many SQLite commands can be executed as options for the sqlite3 command, and then they don't begin with a dot but with a dash, like in "sqlite3 -header -column" which is equivalent to .header ON and .mode column on the SQLite shell.

Is this program correct?

Does this program produce the output below?

Note: the SQLite settings are `.header ON` and `.mode column`

```
SELECT 1+1 AS sum;
```

```
sum
---
2
```

TRUE

Feedback: SELECT evaluates the expression and prints it in a table whose header is renamed to "sum". Without the AS statement, the default header would be "1+1". Try it yourself!

What does the `AND` keyword do in a `SELECT` command?

TRUE:

- Combine multiple conditions

FALSE:

- Combine multiple rows and columns
- Combine numeric and text values
- Combine values within a specified list

Feedback: For example in the following command where the rows of two columns of the "people" table is filtered by two conditions both of which must be fulfilled to return the desired results - all people whose `fname` is `Joe`, and whose `lname` is `Jones`.

```
SELECT fname, lname
FROM people
WHERE lname="Jones" AND fname="Joe"
```

SQL commands (like `SELECT`) must be entered in UPPERCASE letters

FALSE

Feedback: The SQL parser does not distinguish lower or upper case letters, except for variables. In tutorials, the SQL commands are often printed in UPPERCASE to distinguish the keywords from the user-defined parts of a command sequence - this is necessary because SQL is a declarative natural programming language, that is it uses plain English words as keywords.

On the SQLite shell, SQLite commands (like `.show`) must be entered in lowercase letters

TRUE

Feedback: by convention, yes - at least on the console, and also not all options - e.g. both `.header ON` and `.header on` work. This is just a rule. The reason is perhaps to keep SQLite as small as possible - too many choices not directly related to the data keeping would bloat the software.

Complete the code to return the output

Before picking an answer, make sure you understand exactly what this query is trying to do.

```
SELECT company_name, founding_year
FROM companies
WHERE founding_year ??? 2010 AND 2012
LIMIT 2;
```

Output:

company$_{name}$	founding year
Root	2012
Software	2011

TRUE:

- BETWEEN

FALSE:

- LIKE
- COUNT
- IN

Feedback: The query looks for the `company_name` with the `founding_year` between 2010 and 2012. `BETWEEN` allows selection of a range of values. By contrast, with `IN` you can specify a list of potential values `(value1, value2...)`, and with `LIKE` you can search for a pattern using wildcards, like `'A%'` for all values starting with an `A`. `COUNT` is an aggregate function that cannot be used in a `WHERE` filter condition.

What does `GROUP BY` do?

If in doubt, don't guess, but draw a table and play the command out with an example!

TRUE:

- Groups results by one or more columns

FALSE:

- Filters results by one or more columns
- Filters results according to the values of one or more columns
- Filters results based on the results of an aggregate function

Feedback: E.g. in a table with surnames, the following command would count all same surnames and print them in two columns, `surname` and `count`:

```
SELECT surname, COUNT(*) AS count FROM people GROUP BY surname;
```

Complete the code and to return the output

This query uses the same database, `films`, as the DataCamp lesson. However, any other (numeric) column would also have worked.

```
SELECT MAX(duration) [1]  [2]
FROM films;
```

Output:

$\text{max}_{\text{duration}}$
334

You need to select two answers to answer this question.

TRUE:

- `[1] AS [2] max_duration`

FALSE:

- `[1] ALIAS [2] max_duration`
- `[1] == [2] max_duration`
- `[1] IS [2] 334`

Feedback: The alias operator is `AS` in SQL, not `ALIAS`. `==` is a logical operator to test for equality, and there is also no `IS` operator.

Complete the code to return the output.

Get the number of German language films.

```
SELECT [1]
FROM films
WHERE language [2] [3]
```

Output:

count
44

TRUE:

- `[1] COUNT(*) AS count [2] = [3] 'German'`

FALSE:

- `[1] count [2] = [3] 'german'`
- `[1] COUNT(*) [2] == [3] 'German'`
- `[1] COUNT(*) [2] IS [3] 'German'`

Feedback: In DataCamp, you use PostgreSQL, where `COUNT(*)` is printed as `count` without the `AS` (unlike in SQLite). - `SELECT count` queries the column `count`, and does not count. Also, we're looking for `'German'`, not `'german'` - values of string variables are case sensitive. - IS is not an operator.

Author: Marcus Birkenkrahe
Created: 2022-02-27 Sun 22:00
Validate