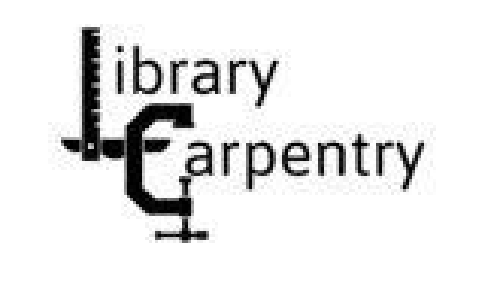
LIBRARY CARPENTRY

SQL



University of North Texas Libraries
Friday, August 20, 2021

Complete the pre-workshop survey:
https://carpentries.typeform.com/to/wi32rS?slug=2021-08-06-unt-online

Link to material covered today:
https://librarycarpentry.org/lc-sql/

Workshop attendees are expected to follow the Carpentries code of conduct:
https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

Setup

Before You Start:

You will need to install **DB Browser for SQLite** and download the **doaj-article-sample** database. See Setup for instructions and further information.

- DB Browser for SQLite download page
  https://sqlitebrowser.org/dl/

- Library Carpentry: SQL setup page
  https://librarycarpentry.org/lc-sql/setup.html

Download the data

To import data, you'll need to open DB Browser for SQLite and download a zip file containing the data files for this tutorial.

1. Download the data files doaj-article-sample.zip from Zenodo.
2. Open the zip file with the zip utlity on your machine and save the folder and files to a location where you can easily find them. For example, your Desktop.
3. Contained in the zip file are two files, doaj-article-sample.db and doaj-article-sample.db.sql. You can either open the database file (less steps) or import the SQL file (more steps).

https://zenodo.org/record/2822005#.YRqN3YhKhaQ

INTRODUCTIONS

- Name, pronouns if you choose, location
- One thing you would like to automate in your work and/or what you hope to learn today

THIS IS ME:

Sarah Lynn Fisher, she/her, sarahlynn.fisher@unt.edu

University of North Texas Libraries

Program Coordinator – Digital Newspaper Unit

- National Digital Newspaper Program – Texas
- Texas Digital Newspaper Program
- Gateway to Oklahoma History

Automate microfilm metadata database

WHY LIBRARY CARPENTRY?

Library Carpentry workshops teach people working in library- and information-related roles how to:

- Cut through the jargon terms and phrases of software development and data science and apply concepts from these fields in library tasks;
- Identify and use best practices in data structures;
- Learn how to programmatically transform and map data from one form to another;
- Work effectively with researchers, IT, and systems colleagues;
- Automate repetitive, error prone tasks.

WORKSHOP GOALS

- Teach skills
- Get started and introduce what’s possible
- Build confidence in using these skills
- Encourage people to continue learning
- Positive learning experience

WORKSHOP REMINDERS

- Raise your hand or post questions in chat and the instructor or a helper will assist you
- Mute yourself if you aren’t speaking
- Post comments in chat - we will do our best to read these out loud at regular intervals
- We will take a break at 10:30 a.m. CST

INTRODUCTION TO SQL

- **S**tructured **Q**uery **L**anguage, or SQL
- Used to communicate with relational databases
- Performs tasks like updating or retrieving data – optimized for handling large amounts of data
- Not a general programming language – keeps data separate from analysis
- SQL *queries* can be called from programming languages, like Python, to interact with databases
- Several variants – all support the same basic statements

RELATIONAL DATABASES

- Consist of one or more tables of data
- Tables have *fields* (columns) and *records* (rows)
- Every field has a data *type*
- Every value in the same field of each record has the same type
- Tables are linked via matching fields, e.g. Title
- *Queries* are commands that find information or make calculations

DATABASE MANAGEMENT SYSTEMS

- SQL is the standard language for relational database management systems
  - Common examples: MySQL, MS Access, SQL Server, Oracle, Filemaker Pro
- SQLite – what we will use in today's workshop
- Differences are only details of import/export of data and datatypes, proprietary extensions
- First step in building custom web applications that serve data to uses
  - WordPress, ecommerce sites like Amazon run on an SQL databases

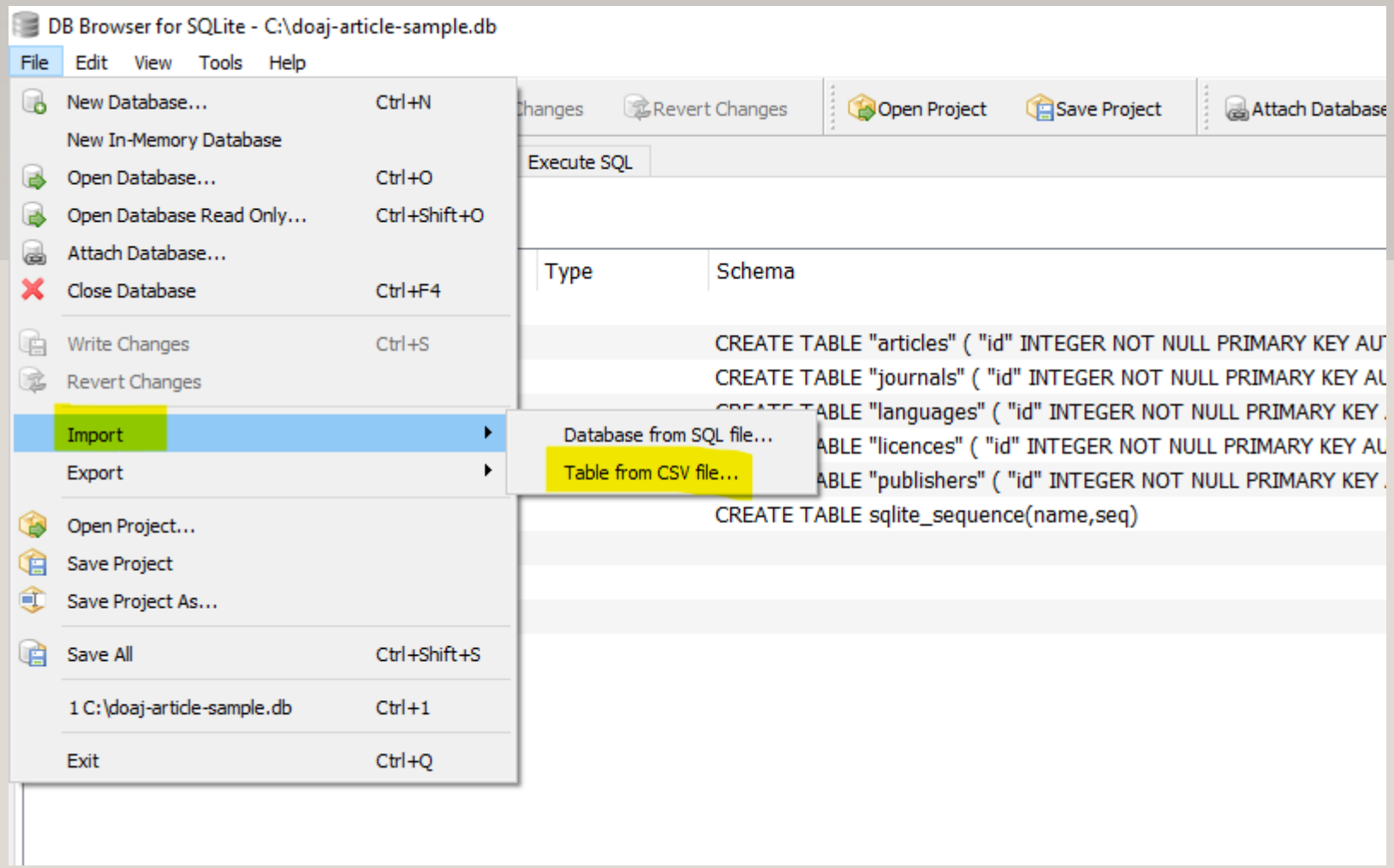
INTRODUCTION TO DB BROWSER FOR SQLite

Open the database file

- Open DB Browser for SQLite
- Choose “File” > “Open Database” from the menu bar at the top of your screen.
- Navigate to where you saved the doaj-article-sample folder and/or files. For example, your Desktop.
- Select “doaj-article-sample.db”.

INTRODUCTION TO DB BROWSER FOR SQLite

- See Tables on left side of the screen
- To see contents of table, click on the table and click Browse Data
- Write a query, click on Execute SQL tab
- Two ways to add new data:
  - Enter data into a CSV file and append
    - Adding data from a CSV file: Choose “File” > “Import” > “Table” from CSV file
  - Click the “Browse Data” tab, then click the “New Record” button


DB Browser for SQLite - C:\doaj-article-sample.db
File
Edit
View
Tools
Help
New Database... Ctrl+N
New In-Memory Database
Open Database... Ctrl+O
Open Database Read Only... Ctrl+Shift+O
Attach Database...
Close Database Ctrl+F4
Write Changes Ctrl+S
Revert Changes
Import
Export
Open Project...
Save Project
Save Project As...
Save All Ctrl+Shift+S
1 C:\doaj-article-sample.db Ctrl+1
Exit Ctrl+Q
Database from SQL file...
Table from CSV file...
Revert Changes
Open Project
Save Project
Attach Database
Execute SQL
Type
Schema
CREATE TABLE "articles" ("id" INTEGER NOT NULL PRIMARY KEY AU
CREATE TABLE "journals" ("id" INTEGER NOT NULL PRIMARY KEY AU
ABLE "licences" ("id" INTEGER NOT NULL PRIMARY KEY AU
ABLE "publishers" ("id" INTEGER NOT NULL PRIMARY KEY
CREATE TABLE sqlite_sequence(name,seq)

DB Browser for SQLite - C:\doaj-article-sample.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Re Changes | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

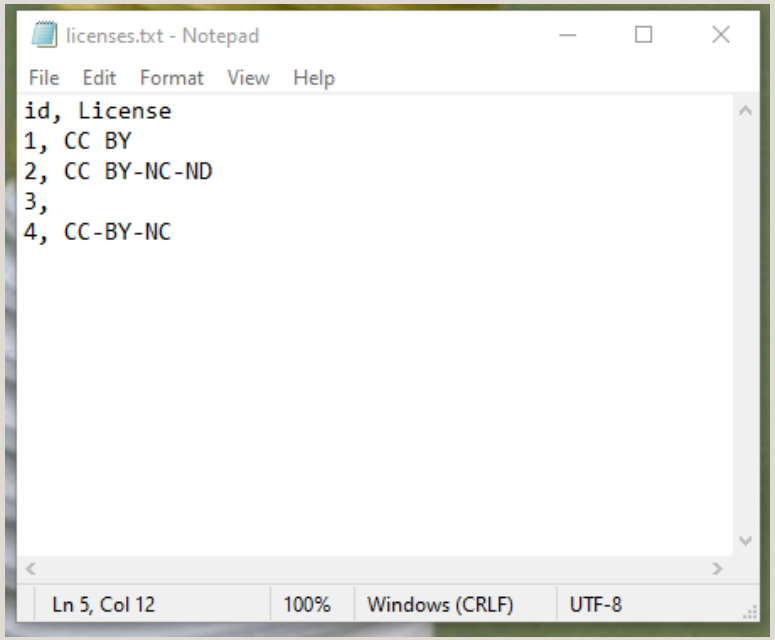
Table: articles | Filter in any column

Insert a new record in the current table

	id	Title		DOI	URL	
	Filter	Filter	Filter	Filter	Filter	Filter
1	0	The Fisher Thermodynamics of Quasi-Probabilities	Flavia Pennini|Angelo Plastino	10.3390/e17127853	https://doaj.org/article/...	Fisher information|
2	1	Aflatoxin Contamination of the Milk Supply: A ...	Naveed Aslam|Peter C. Wynn	10.3390/agriculture5041172	https://doaj.org/article/...	aflatoxins|AFM1|AF
3	2	Metagenomic Analysis of Upwelling-Affected ...	Rafael R. C. Cuadrat|Juliano C. Cury|Alberto M. ...	10.3390/ijms161226101	https://doaj.org/article/...	PKS|NRPS|metager
4	3	Synthesis and Reactivity of a Cerium(III) ...	Fabrizio Ortu|Hao Zhu|Marie-Emmanuelle Boulo...	10.3390/inorganics3040534	https://doaj.org/article/...	lanthanide|cerium|
5	4	Performance and Uncertainty Evaluation of Snow...	Magali Troin|Richard Arsenault|François Brissette	10.3390/hydrology2040289	https://doaj.org/article/...	snow models|hydro
6	5	Dihydrochalcone Compounds Isolated from ...	Xiaoxiao Qin|Yun Feng Xing|Zhiqin Zhou|Yuncon...	10.3390/molecules201219754	https://doaj.org/article/...	Malus crabapples|le
7	6	Ionic Liquids as Carbene Catalyst Precursors in t...	Anton Axelsson|Linda Ta|Henrik Sundén	10.3390/catal5042052	https://doaj.org/article/...	ionic liquid|NHC|OT
8	7	Characterization of Aspartate Kinase from ...	Weihong Min|Huiying Li|Hongmei Li|Chunlei Liu|...	10.3390/ijms161226098	https://doaj.org/article/...	Corynebacterium pe
9	8	Quaternifications and Extensions of Current ...	Tosiaki Kori|Yuto Imai	10.3390/sym7042150	https://doaj.org/article/...	infinite dimensional
10	9	Imaging of HCC—Current State of the Art	Christina Schraml|Sascha Kaufmann|Hansjoerg ...	10.3390/diagnostics5040513	https://doaj.org/article/...	hepatocellular carci
11	10	Synthesis and Complexation of Well-Defined ...	Mark Billing|Tobias Rudolph|Eric Täuscher|Raine...	10.3390/polym7121526	https://doaj.org/article/...	atom transfer radic
12	11	UAV Control on the Basis of 3D Landmark Bearin...	Simon Karpenko|Ivan Konovalenko|Alexander ...	10.3390/s151229768	https://doaj.org/article/...	UAV|visual odometr
13	12	Polar Glycosylated and Lateral Non-Glycosylated ...	Kelly M. Fulton|Elena Mendoza-Barberá|Susan M...	10.3390/ijms161226097	https://doaj.org/article/...	O-flagellin polar gly
14	13	Selective Oxidation of Glycerol with 3% H2O2 ...	Gongde Wu|Xiaoli Wang|Taineng Jiang|Qibo Lin	10.3390/catal5042039	https://doaj.org/article/...	layered-double hydr
15	14	A Copper-Based Metal-Organic Framework as an...	Wei Long|Wenge Qiu|Chongwei Guo|Chuanqian...	10.3390/molecules201219756	https://doaj.org/article/...	metal-organic fram
16	15	Performance-Based Cognitive Screening ...	Andrew J. Larner	10.3390/diagnostics5040504	https://doaj.org/article/...	diagnosis|cognitive
17	16	Aberrant GLI1 Activation in DNA Damage ...	Komaraiah Palle|Chinnadurai Mani|Kaushlendra ...	10.3390/cancers7040894	https://doaj.org/article/...	hedgehog signaling
18	17	Trends and Potential of the Market for Combine ...	Clemens Fuchs|Joachim Kasten|Mathias Urbanek	10.3390/machines3040364	https://doaj.org/article/...	combine harvesters

DATASET DESCRIPTION

- 5 CSV (comma-separate values) files: articles, journals, languages, licenses, publishers
- The information in these tables are from a sample of 51 different journals published during 2015.
- CSV - plain text file that stores tables and spreadsheet



DB Browser for SQLite - C:\doaj-article-sample.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: licences

	id	Licence
	Filter	Filter
1	1	CC BY
2	2	CC BY-NC-ND
3	3	
4	4	CC BY-NC

DATASET DESCRIPTION

articles

- Contains individual article Titles and the associated citations and metadata
- (16 fields, 1001 records)
- Field names: id, Title, Authors, DOI, URL, Subjects, ISSNs, Citation, LanguageID, LicenseID, Author_Count, First_Author, Citation_Count, Day, Month, Year

journals

- Contains various journal Titles and associated metadata.
- (5 fields, 51 records)
- Field names: id, ISSN-L,ISSNs, PublisherID, Journal_Title

languages

- ID table which associates language codes with id numbers
- (2 fields, 4 records)
- Field names: id, Language

licenses

- ID table which associates License codes with id numbers
- (2 fields, 4 records)
- Field names: id, Licence

publishers

- ID table which associates Publisher names with id numbers
- (2 fields, 6 records)
- Field names: id, Publisher

SQL Data Type Quick Reference

Different database software/platforms have different names and sometimes different definitions of data types, so you'll need to understand the data types for any platform you are using. The following table explains some of the common data types and how they are represented in SQLite; more details available on the SQLite website.

Data type	Details	Name in SQLite
boolean or binary	this variable type is often used to represent variables that can only have two values: yes or no, true or false.	doesn't exist - need to use integer data type and values of 0 or 1.
integer	sometimes called whole numbers or counting numbers. Can be 1,2,3, etc., as well as 0 and negative whole numbers: -1,-2,-3, etc.	INTEGER
float, real, or double	a decimal number or a floating point value. The largest possible size of the number may be specified.	REAL
text or string	and combination of numbers, letters, symbols. Platforms may have different data types: one for variables with a set number of characters - e.g., a zip code or postal code, and one for variables with an open number of characters, e.g., an address or description variable.	TEXT
date or datetime	depending on the platform, may represent the date and time or the number of days since a specified date. This field often has a specified format, e.g., YYYY-MM-DD	doesn't exist - need to use built-in date and time functions and store dates in real, integer, or text formats. See Section 2.2 of SQLite documentation for more details.
blob	a Binary Large OBject can store a large amount of data, documents, audio or video files.	BLOB

Highlighted are main data types in our doaj-article-sample database.

MINI BREAK

Description: dog wearing sunglasses

SELECTING AND SORTING DATA

- A *Query* is a question or request for data
- Query a database by asking the same question using a common language - SQL in this case
- Queries can have multiple *statements*
- SQL statement types can be grouped into five different categories:
    - Data definition language (DDL)
    - Data manipulation language (DML) – Managing data within tables
    - Data Control Language (DCL)
    - Transaction Control Statement (TCS)
    - Session Control Statements (SCS)

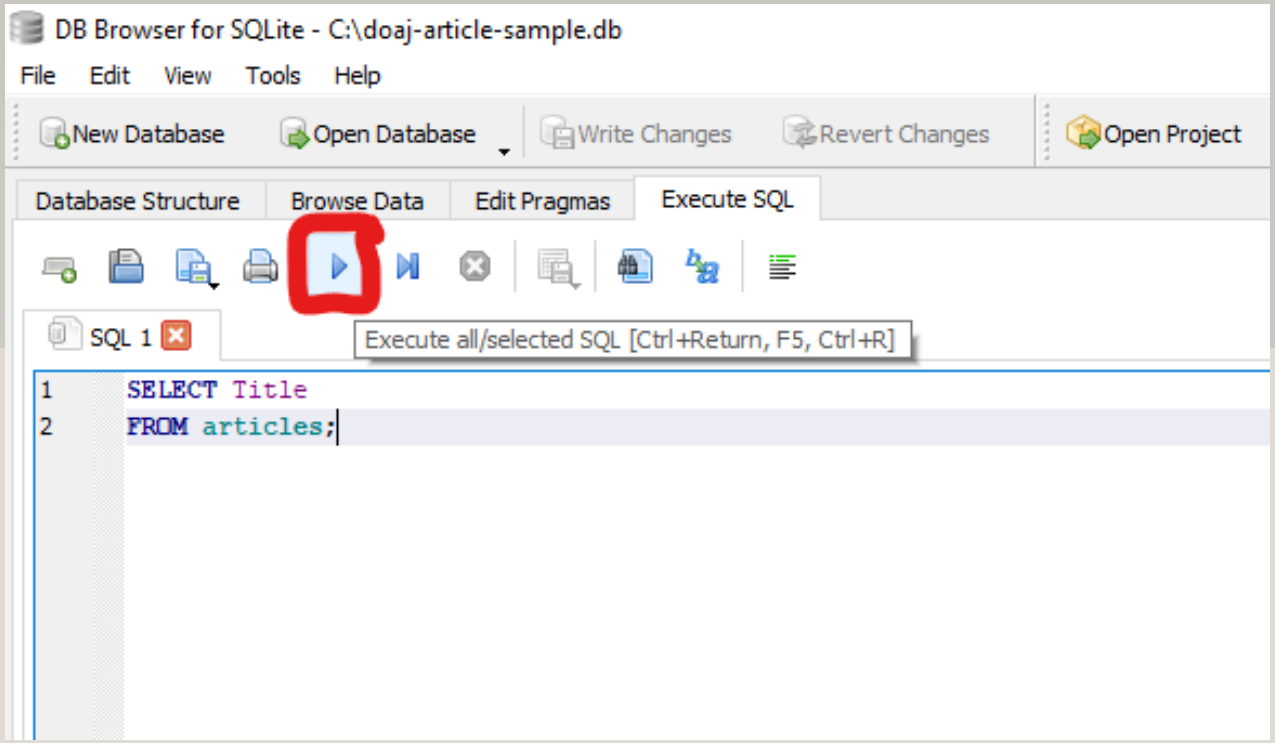
WRITE YOUR FIRST QUERY

Write a SQL query that selects only the "Title" column from the "articles" table.

#1

SQL

```
SELECT title
FROM articles;
```



- Capitalization of SELECT and FROM is only for readability and represents good style
- Some tables and columns require capitalization and some do not
- Final statement in query should end with a semi-colon (;)

GOOD STYLE

- Many variations in SQL queries
- Good style makes your queries easier to read
- Choose column names that are short (one or two words) when designing your tables
- Spaces in column names will create errors, use CamelCase or An_Underscore
- SQL keywords/commands are case-insenstitive, but it matters in some SQL programs. Check capitalization if your query isn’t working.

READABILITY

- General consensus with SQL - if you can break it into components on multiple lines, queries become easier to read

SQL

```
SELECT articles.Title, articles.First_Author, journals.Journal_Title, publishers.Publisher FROM articles JOIN journals ON articles.ISSNs = journals.ISSNs J
OIN publishers ON publishers.id = journals.PublisherId;
```

SQL

```
SELECT articles.Title, articles.First_Author, journals.Journal_Title, publishers.Publisher
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
JOIN publishers
ON publishers.id = journals.PublisherId;
```

MORE SELECTION QUERIES

- Add more fields to retrieve more information

#2

```sql
SELECT Title, Authors, ISSNs, Year
FROM articles;
```

- Use wildcard (*) to select all of the columns in a table

#3

```sql
SELECT *
FROM articles;
```

UNIQUE VALUES

- Add DISTINCT to find only unique records

#4

```sql
SELECT DISTINCT ISSNs
FROM articles;
```

- If we select more than one column, then the distinct pairs of values are returned.

#5

```sql
SELECT DISTINCT ISSNs, Day, Month, Year
FROM articles;
```

SORTING

- Sort the results of queries using the keyword ORDER BY, also referred to as a *clause*

Create a query that sorts the articles table in ascending order by ISSNs using the ASC keyword in conjunction with ORDER BY.

#6

SQL

```sql
SELECT *
FROM articles
ORDER BY ISSNs ASC;
```

- ASC is the default

SORTING

- Add DESC to sort in a different direction

#7

SQL

```sql
SELECT *
FROM articles
ORDER BY First_Author DESC;
```

- Sort several fields in different directions

#8

SQL

```sql
SELECT *
FROM articles
ORDER BY ISSNs DESC, First_Author ASC;
```

CHALLENGE

#8

Challenge

Write a query that returns `Title`, `First_Author`, `ISSNs` and `Citation_Count` from the articles table, ordered by the top cited article and alphabetically by title.

Solution

SQL

```sql
SELECT Title, First_Author, ISSNs, Citation_Count
FROM articles
ORDER BY Citation_Count DESC, Title ASC;
```

MINI BREAK

Description: Monkey hugging a kitten

https://gateway.okhistory.org/ark:/67531/metadc427038/

FILTERING

- Find data for a specific set of conditions using the WHERE clause

Write a query that returns only articles from the journal Acta Crystallographica (ISSN 2056-9890).

#10

SQL

```sql
SELECT *
FROM articles
WHERE ISSNs='2056-9890';
```

FILTERING

- Add additional conditions using AND, OR, and/or NOT in clauses

11: Write a query that returns only articles from the journal Acta Crystallographica (ISSN 2056-9890) published after October. 12: Write a query that returns articles from Humanities and *Religions journals (ISSNs* "2076-0787" and "2077-1444")

#11

SQL

```
SELECT *
FROM articles
WHERE (ISSNs='2056-9890') AND (Month > 10);
```

#12

SQL

```
SELECT *
FROM articles
WHERE (issns = '2076-0787') OR (issns = '2077-1444');
```

- Parentheses are used merely for readability in this case but can be required by the SQL interpreter in order to disambiguate formulas.

FILTERING

- Use comparison keywords; such as LIKE, IN, BETWEEN … AND, IS NULL; when you do know the value you are searching for.

Write a query that returns all of the data where the subject contains "Crystal Structure."

#13

```sql
SELECT *
FROM articles
WHERE Subjects LIKE '%Crystal Structure%';
```

- The wildcard character "%" is used to match zero to many characters.

CHALLENGE

#14

Challenge

Write a query that returns the `Title`, `First_Author`, `Subjects`, `ISSNs`, `Month` and `Year` for all papers where `Subjects` contains “computer” and that have more than 8 citations.

Solution

SQL

```sql
SELECT Title, First_Author, Subjects, ISSNs, Month, Year
FROM articles
WHERE (Subjects LIKE '%computer%') AND (Citation_Count > 8);
```

MINI BREAK

Description: circus ponies

https://gateway.okhistory.org/ark:/67531/metadc1474023/

ORDERING

Order of execution

#14

```sql
SELECT Title, Authors
FROM articles
WHERE ISSNs = '2067-2764|2247-6202'
ORDER BY First_Author ASC;
```

📌 The computer is basically doing this:

1. Filtering rows according to WHERE
2. Sorting results according to ORDER BY
3. Displaying requested columns or expressions.

- It isn’t necessary to display the First_Author column in order to sort by it
- Sorting occurs earlier in the computational pipeline
- Clauses are written in a fixed order: SELECT, FROM, WHERE, then ORDER BY

COMPLEX QUERIES

- SQL offers the flexibility of iteratively adding new conditions, but this can make them difficult to read and inefficient.

- Make complex queries more readable by:

  1. Rewriting so the logic is easy to follow

  2. Adding comments for context and clarity

- Start with a simple query, add clauses one by one to test effectiveness

COMPLEX QUERIES

#15

```sql
SELECT *
FROM articles
WHERE (ISSNs = '2076-0787') OR (ISSNs = '2077-1444') OR (ISSNs = '2067-2764|2247-6202');
```

- Queries 15 and 16 return the same results; adding IN to the WHERE clause improves readability

#16

```sql
SELECT *
FROM articles
WHERE (ISSNs IN ('2076-0787', '2077-1444', '2067-2764|2247-6202'));
```

COMMENTING

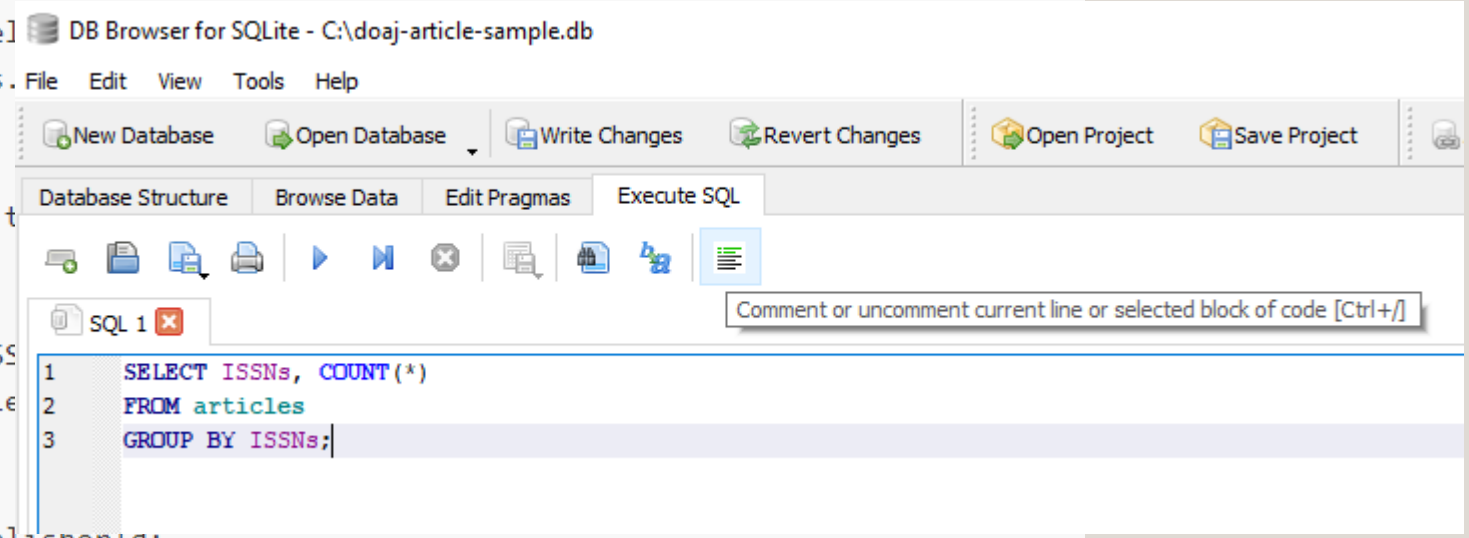
- Comments explain the logic of a query and do not affect the executable code
- Comments begin using – and end at the end of the line
- Enclose a line in /* and */ to make an entire paragraph a comment

#17

SQL

```
/*In this section we want to give an example how to
join multiple tables.*/

-- First we mention all the fiel
SELECT articles.Title, articles.
-- from the first table
FROM articles
-- and join it with the second t
JOIN journals
-- The related attributes are:
ON articles.ISSNs = journals.ISS
-- We want to join a third table
JOIN publishers
-- the related attributes are:
ON publishers.id = journals.Publisneria;
```



BREAK (10 minutes)

Description: Squirrel relaxing on bench on UNT Library Mall on a hot day

AGGREGATING AND CALCULATING VALUES

- Most common functions are MAX, MIN, AVG, COUNT, SUM
  - MAX (find the maximum value in a field)
  - MIN (find the minimum value in a field)
  - AVG (find the average value of a field)
  - COUNT (count the number of values in a field and present the total)
  - SUM (add up the values in a field and present the sum).

AGGREGATING AND CALCULATING VALUES

Write a query that returns the average Citation_Count for each journal in "articles".

#18

SQL

```sql
SELECT ISSNs, AVG(Citation_Count)
FROM articles
GROUP BY ISSNs;
```

- GROUP BY is used by SQL to arrange identical data into groups
- This process is also called *aggregation* – combining results based on value and calculating combined values in groups

AGGREGATING AND CALCULATING VALUES

#19

```sql
SELECT ISSNs, AVG(Citation_Count)
FROM articles
GROUP BY ISSNs
ORDER BY AVG(Citation_Count) DESC;
```

- Use ORDER BY clause to make results more useful

AGGREGATING AND CALCULATING VALUES

Challenge

Write a query using an aggregate function that returns the number of article titles per ISSNs, sorted by title count in descending order. Which ISSN has the most titles? (Hint to choosing which aggregate function to use - it is one of the common aggreggate functions `MAX, MIN, AVG, COUNT, SUM`.)

Solution

SQL

```sql
SELECT ISSNs, COUNT(Title)
FROM articles
GROUP BY ISSNs
ORDER BY count(Title) DESC;
```

AGGREGATING AND CALCULATING VALUES

- *Having* keyword filters results based on aggregate functions – works like *WHERE* clause

#20

```sql
SELECT ISSNs, COUNT(*)
FROM articles
GROUP BY ISSNs
HAVING count(Title) >= 10;
```

- Query #20 returns only information about journals with 10 or more published articles

- *Having* appears after GROUP BY statement: the data are retrieved (SELECT), can be filtered (WHERE), then joined in groups (GROUP BY); finally, we only select some of these groups (HAVING).

AGGREGATING AND CALCULATING VALUES

Challenge

Write a query that returns, from the `articles` table, the average `Citation_Count` for each journal ISSN but only for the journals with 5 or more citations on average.

Solution

SQL

```
SELECT ISSNs, AVG(Citation_Count)
FROM articles
GROUP BY ISSNs
HAVING AVG(Citation_Count)>=5;
```

CALCULATIONS

- *Computed columns* are calculations within queries

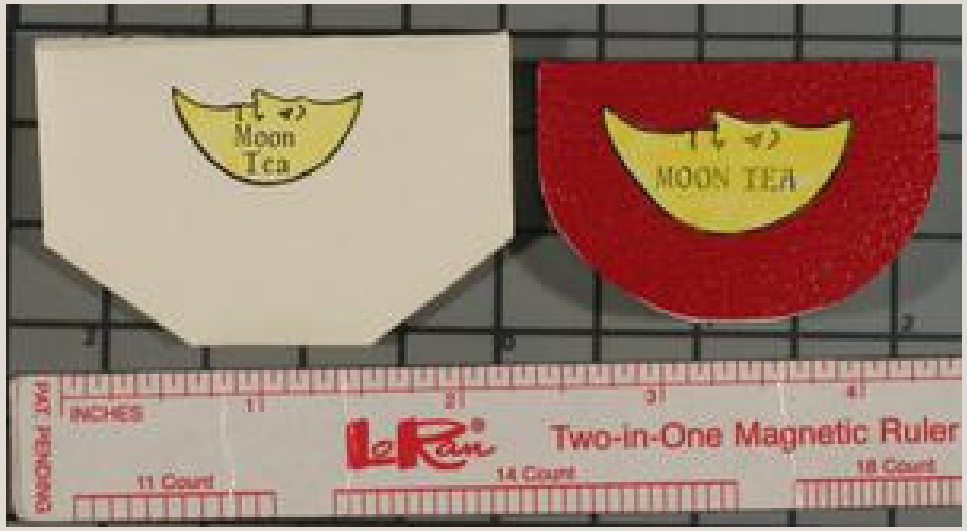
#21

```sql
SELECT Title, ISSNs, Author_Count -1 as CoAuthor_Count
FROM articles
ORDER BY Author_Count -1 DESC;
```

- Expressions on a column or columns get new values during a query

- Arithmetic operators (like +, -, *, /, square root SQLRT or the modulo operator %) are also useful for calculating new values

MINI BREAK



Description: photograph of miniature book and ruler

https://texashistory.unt.edu/ark:/67531/metadc3588/

JOINS AND ALIASES

- JOIN clause allows us to combine columns from one or more tables in a database by using values common to each
- Follows the FROM clause in a SQL statement
- Tell the computer which columns provide the link between the two tables using the word ON
- Creating aliases allows us to spend less time typing, and more time querying!

JOINS AND ALIASES

Write a query that joins the articles table with the journals table.

#22

```sql
SELECT *
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs;
```

- ISSNs columns in both these tables link them
- ON, like WHERE filters, according to a test condition

#23

```sql
SELECT *
FROM articles
JOIN journals
USING (ISSNs);
```

- Use the table.column format or the word USING

JOINS AND ALIASES

- Use table.colname to join in the SELECT clause

#24

SQL

```
SELECT articles.ISSNs, journals.Journal_Title, articles.Title, articles.First_Author, articles.Month, articles.Year
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs;
```

JOINS AND ALIASES

- Joins can be combined with sorting, filtering, and aggregation

#25

SQL

```sql
SELECT articles.ISSNs, journals.Journal_Title, ROUND(AVG(articles.Author_Count), 2)
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
GROUP BY articles.ISSNs;
```

- ROUND function rounds the Author_Count number returned by the AVG function by 2 decimal places.

JOINS AND ALIASES

Challenge

Write a query that `JOINS` the `articles` and `journals` tables and that returns the `Journal_Title`, total number of articles published and average number of citations for every journal ISSN.

Solution

SQL

```
SELECT journals.Journal_Title, count(*), avg(articles.Citation_Count)
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
GROUP BY articles.ISSNs;
```

JOINS AND ALIASES

- You can also JOIN multiple tables:

#26

SQL

```sql
SELECT articles.Title, articles.First_Author, journals.Journal_Title, publishers.Publisher
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
JOIN publishers
ON publishers.id = journals.PublisherId;
```

JOINS AND ALIASES

Challenge:

Write a query that returns the `Journal_Title`, `Publisher` name, and number of articles published, ordered by number of articles in descending order.

Solution

SQL

```
SELECT journals.Journal_Title, publishers.Publisher, COUNT(*)
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
JOIN publishers
ON publishers.id = journals.PublisherId
GROUP BY Journal_Title
ORDER BY COUNT(*) DESC;
```

JOINS AND ALIASES

- To make things clearer as queries become more complex, use aliases to assign new names to items in the query

#27 Table names

```sql
SELECT ar.Title, ar.First_Author, jo.Journal_Title
FROM articles AS ar
JOIN journals  AS jo
ON ar.ISSNs = jo.ISSNs;
```

#28 Column names

```sql
SELECT ar.title AS title, ar.first_author AS author, jo.journal_title AS journal
FROM articles AS ar
JOIN journals  AS jo
ON ar.issns = jo.issns;
```

- AS is not required for the query to work but is an example of good style

MINI BREAK

Description: orange and white cat inside a clothes dryer

https://texashistory.unt.edu/ark:/67531/metapth908662/

SAVING QUERIES

- Views are queries saved in the database
- Query a view as a (virtual) table that is populated every time you query it

#29

```sql
CREATE VIEW journal_counts AS
SELECT ISSNs, COUNT(*)
FROM articles
GROUP BY ISSNs;
```

- Saving a query requires you to add CREATE VIEW viewname AS before the query itself

SAVING QUERIES

- Access the results of query #29 using shorter notation:

#30

```sql
SELECT *
FROM journal_counts;
```

- Remove the above view from the database using DROP VIEW

#31

```sql
DROP VIEW journal_counts;
```

SAVING QUERIES

DB Browser for SQLite - C:\doaj-article-sample.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Open P

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1

Export to CSV

Save as view

```
1  SELECT ISSNs, COUNT(*)
2  FROM articles
3  GROUP BY ISSNs;
```

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Create Table | Create Index | Print

Name	Type	Schema
Tables (6)		
articles		CREATE TABLE "articles" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, "Title" TEXT,
journals		CREATE TABLE "journals" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, "ISSN-L" TE
languages		CREATE TABLE "languages" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, "Languag
licences		CREATE TABLE "licences" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, "Licence" TE
publishers		CREATE TABLE "publishers" ("id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, "Publisher
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
Indices (0)		
Views (1)		
journaLcounts		CREATE VIEW "journaLcounts" AS SELECT ISSNs, COUNT(*) FROM articles GROUP BY ISSNs
Triggers (0)		

7	1660-4601	4
8	1996-1944	2

SAVING QUERIES

Challenge

Write a `CREATE VIEW` query that `JOINS` the `articles` table with the `journals` table on `ISSNs` and returns the `COUNT` of article records grouped by the `Journal_Title` in `DESC` order.

Solution

SQL

```
CREATE VIEW journal_counts AS
SELECT journals.Journal_Title, COUNT(*)
FROM articles
JOIN journals
ON articles.ISSNs = journals.ISSNs
GROUP BY Journal_Title
ORDER BY COUNT(*) DESC
```

MINI BREAK

Description: Photograph of Julia Muegge's dog sleeping on a dog bed and covered by a blanket.

https://texashistory.unt.edu/ark:/67531/metadc1752551/

CREATING TABLES AND MODIFYING DATA

- CREATE TABLE
  - Two words for a single command
  - Creates a new table
  - Arguments are names and types of the table's columns

#32

```sql
CREATE TABLE journals(id text, ISSN-L text, ISSNs text, PublisherId text, Journal_Title text);
```

#33

```sql
DROP TABLE journals;
```

- Removes a table
- Use with care!

CREATING TABLES AND MODIFYING DATA

- Can specify several kinds of constraints on its columns when creating a table

#34

SQL

```sql
CREATE TABLE "journals" (
        "id"    INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
        "ISSN-L"        TEXT,
        "ISSNs" TEXT,
        "PublisherId"   INTEGER,
        "Journal_Title" TEXT,
        CONSTRAINT "PublisherId" FOREIGN KEY("PublisherId") REFERENCES "publishers"("id")
);
```

CREATING TABLES AND MODIFYING DATA

- Add, change or remove records using INSERT, UPDATE, and DELETE

#35

SQL

```
INSERT INTO "journals" VALUES (1,'2077-0472','2077-0472',2,'Agriculture');
INSERT INTO "journals" VALUES (2,'2073-4395','2073-4395',2,'Agronomy');
INSERT INTO "journals" VALUES (3,'2076-2616','2076-2616',2,'Animals');
```

#36

SQL

```
CREATE TABLE "myjournals"(Journal_Title text, ISSNs text);
INSERT INTO "myjournals" SELECT Journal_Title, long ISSNs journals;
```

- Query 35 inserts rows into the “journals” table
- Query 36 inserts values into one table directly from another

CREATING TABLES AND MODIFYING DATA

- Use UPDATE to modify existing records – good for correcting typos

#37

```sql
UPDATE journals SET ISSN-L = 2076-2615, ISSNs = 2076-2615 WHERE id = 3;
```

Tell the database which table to update, what the values should be for the fields, and conditions WHERE values should be updated. (Otherwise all will be modified!)

#38

```sql
DELETE FROM journals WHERE Journal_Title = 'Animals';
```

- Use DELETE and WHERE clause to match records to discard

CREATING TABLES AND MODIFYING DATA

Exercise

Write an SQL statement to add the journal “New Journal of Physics” (ISSNs & ISSNs: 1367-2630; publisher: “Institute of Physics (IOP)”) to the table `journals`. You need to add the publisher “IOP” to the table `publishers` as well.

Solution

SQL

```
INSERT INTO "publishers" VALUES (7,'Institute of Physics (IOP)');
INSERT INTO "journals" VALUES (52,'1367-2630','1367-2630',7,'New Journal of Physics');
```

CREATING TABLES AND MODIFYING DATA

Backing Up with SQL

SQLite has several administrative commands that aren't part of the SQL standard. One of them is `.dump`, which prints the SQL commands needed to re-create the database. Another is `.read`, which reads a file created by `.dump` and restores the database. A colleague of yours thinks that storing dump files (which are text) in version control is a good way to track and manage changes to the database. What are the pros and cons of this approach? (Hint: records aren't stored in any particular order.)

Solution

Advantages

- A version control system will be able to show differences between versions of the dump file; something it can't do for binary files like databases
- A VCS only saves changes between versions, rather than a complete copy of each version (save disk space)
- The version control log will explain the reason for the changes in each version of the database

Disadvantages

- Artificial differences between commits because records don't have a fixed order

EXTRA CHALLENGES AND SURVEYS

https://librarycarpentry.org/lc-sql/11-extra-challenges/index.html

*POST WORKSHOP SURVEY: https://carpentries.typeform.com/to/UgVdRQ?slug=2021-08-06-unt-online

*SOFTWARE CARPENTRY SURVEY: https://unt.az1.qualtrics.com/jfe/form/SV_7a1WNVkwwPM8RjU