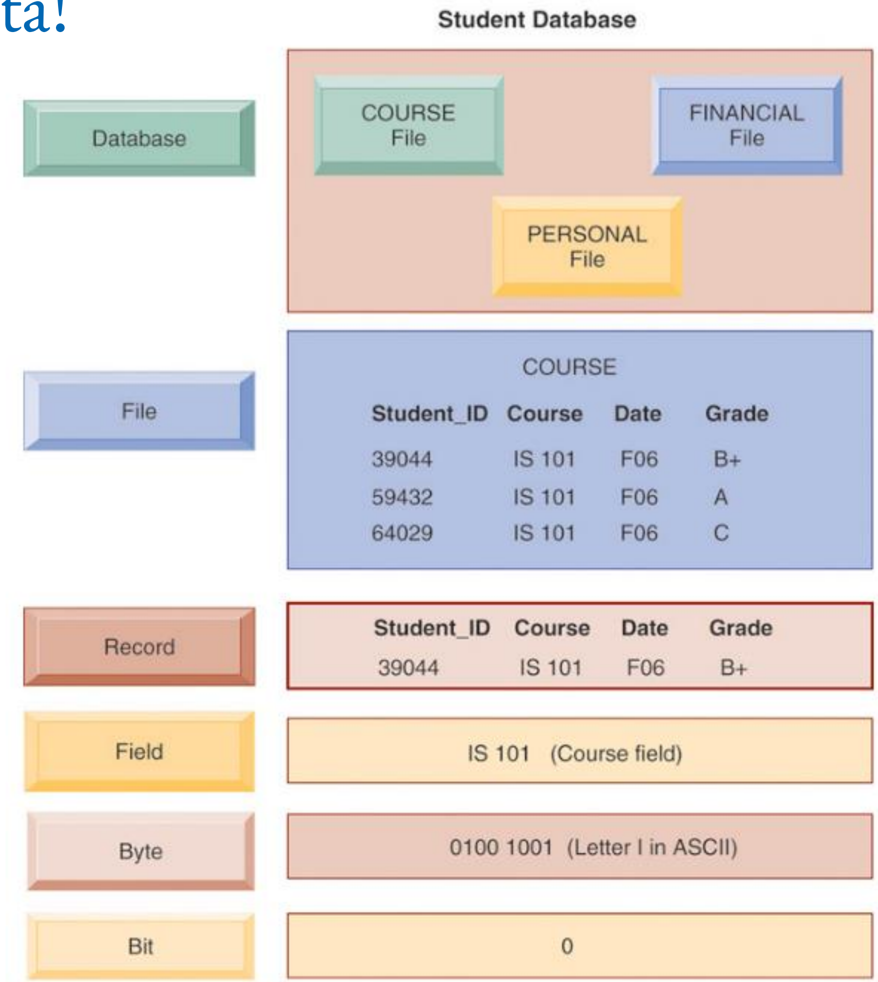
SQLite

CMPSC 305 – Database Systems

Allegheny College

All types of data!



Student Database

Database

COURSE File

FINANCIAL File

PERSONAL File

File

COURSE

Student_ID	Course	Date	Grade
39044	IS 101	F06	B+
59432	IS 101	F06	A
64029	IS 101	F06	C

Record

Student_ID	Course	Date	Grade
39044	IS 101	F06	B+

Field

IS 101 (Course field)

Byte

0100 1001 (Letter I in ASCII)

Bit

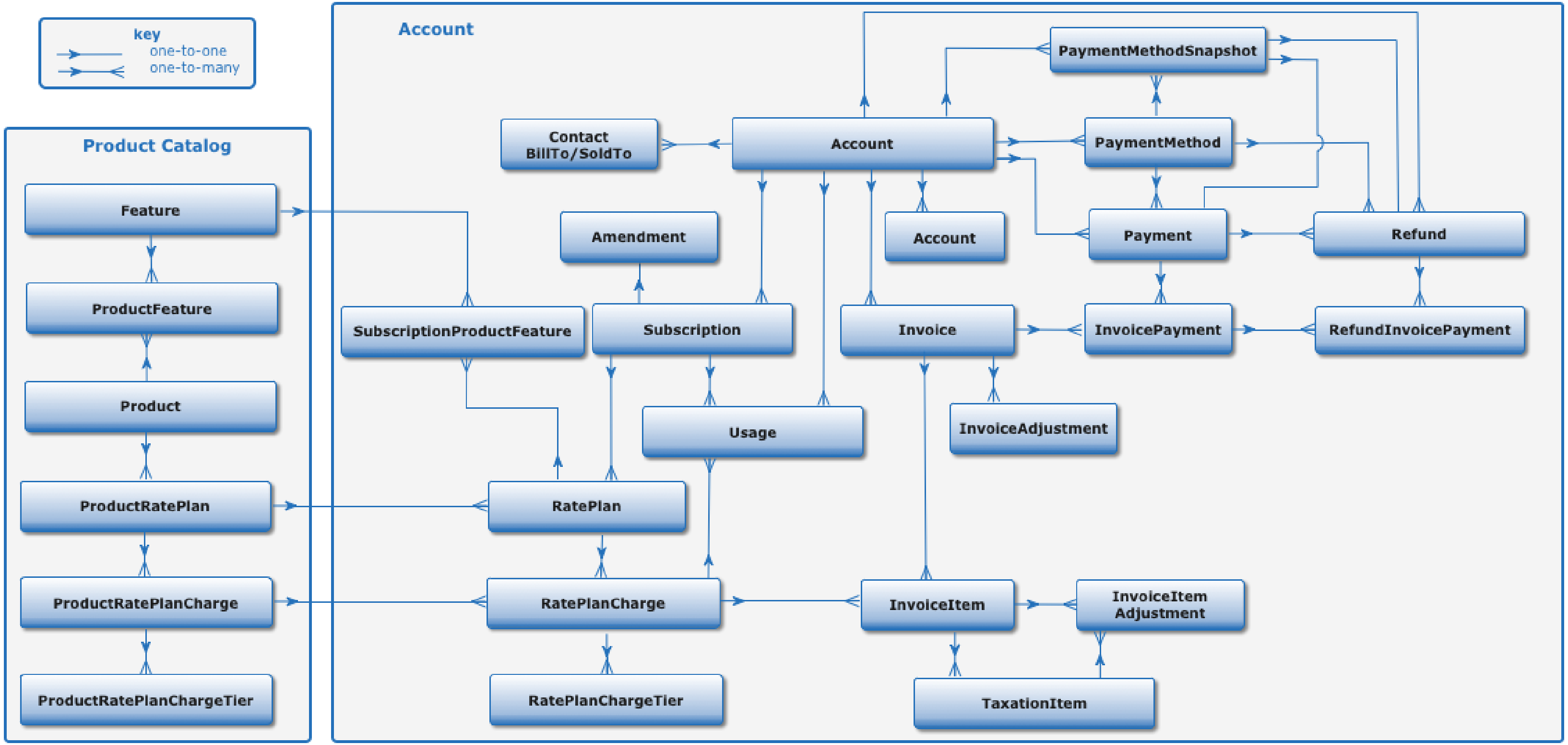
0

A database, simply stated

ID	Dept	RoomNum
JJ	CS	**105**
OBC	CS	**104**
GL	CS	**106**
DL	CS	**112**
HZ	CS	**114**

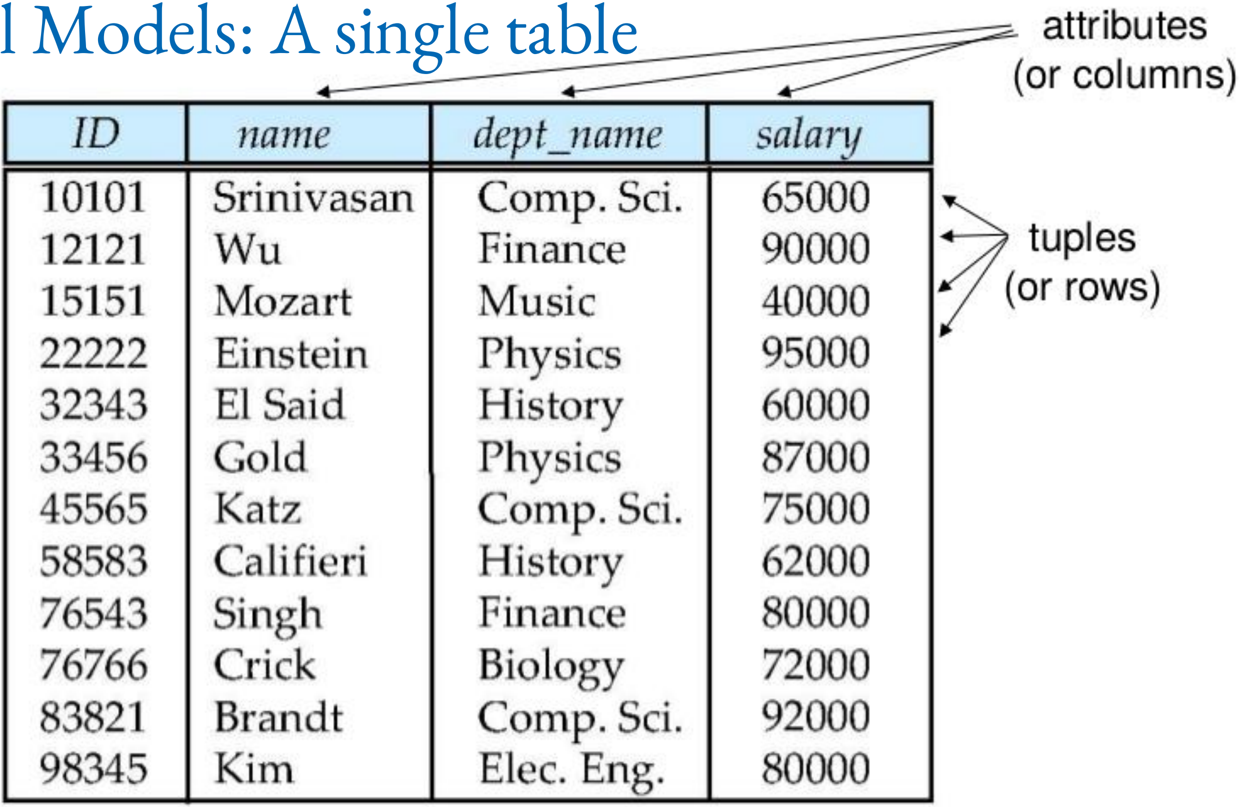
- The entire database fits into one table.
- Is the column "Dept" necessary in this table?

A database, not-so-simply stated



- The entire database is made up of many tables.
- A table must be connected to the others in some way.

Relational Models: A single table

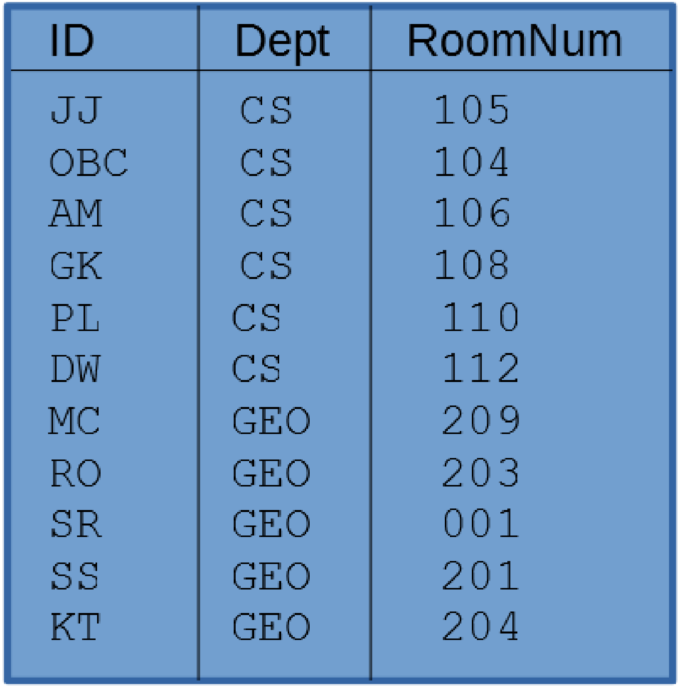


ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

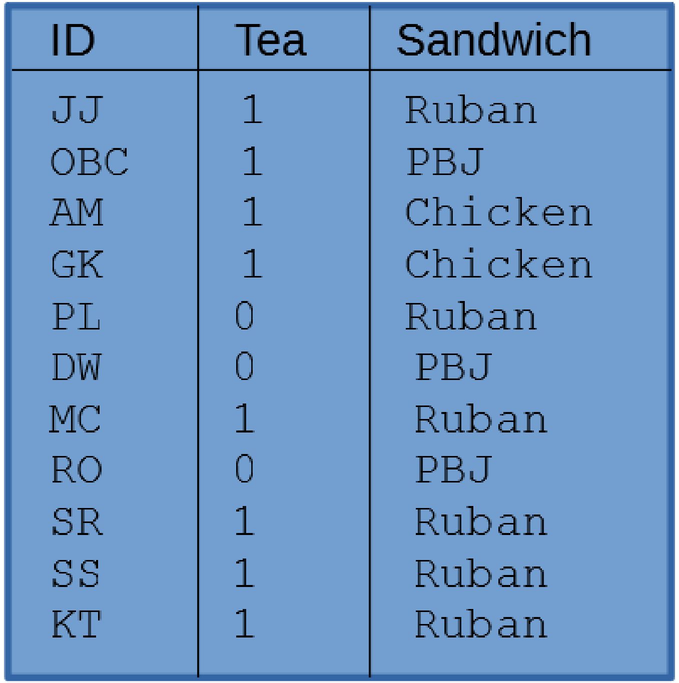
attributes
(or columns)
tuples
(or rows)

- Each field of a row is an "observation"
- Rows are a series (i.e., tuples) of "observations"
- Columns contain same "observation" class (are called attributes)

Specific information for each table



ID	Dept	RoomNum
JJ	CS	105
OBC	CS	104
AM	CS	106
GK	CS	108
PL	CS	110
DW	CS	112
MC	GEO	209
RO	GEO	203
SR	GEO	001
SS	GEO	201
KT	GEO	204



ID	Tea	Sandwich
JJ	1	Ruban
OBC	1	PBJ
AM	1	Chicken
GK	1	Chicken
PL	0	Ruban
DW	0	PBJ
MC	1	Ruban
RO	0	PBJ
SR	1	Ruban
SS	1	Ruban
KT	1	Ruban

- Two tables containing specific types of data, using the same ID on a row
- Each table organizes non-redundant information, but needs a way to connect a row to the rest of the base (i.e., the common ID column serves as a primary key).

E-R Models … Entities? Relationships?

Entities

- The entity-relationship (E-R) data model uses a collection of basic objects, called entities, and relationships exist to connect these objects.
- Entities are defined by attributes (i.e, column headers in tables)
- ID, name, and salary are points of information to describe an instructor entity

Relationships

- A relationship is an association among several entities
- For example, a member relationship associates an instructor with her/his department.
- The set of all entities of the same type and the set of all relationships of the same type are termed an entity set and relationship set, respectively.

Running an SQLite client

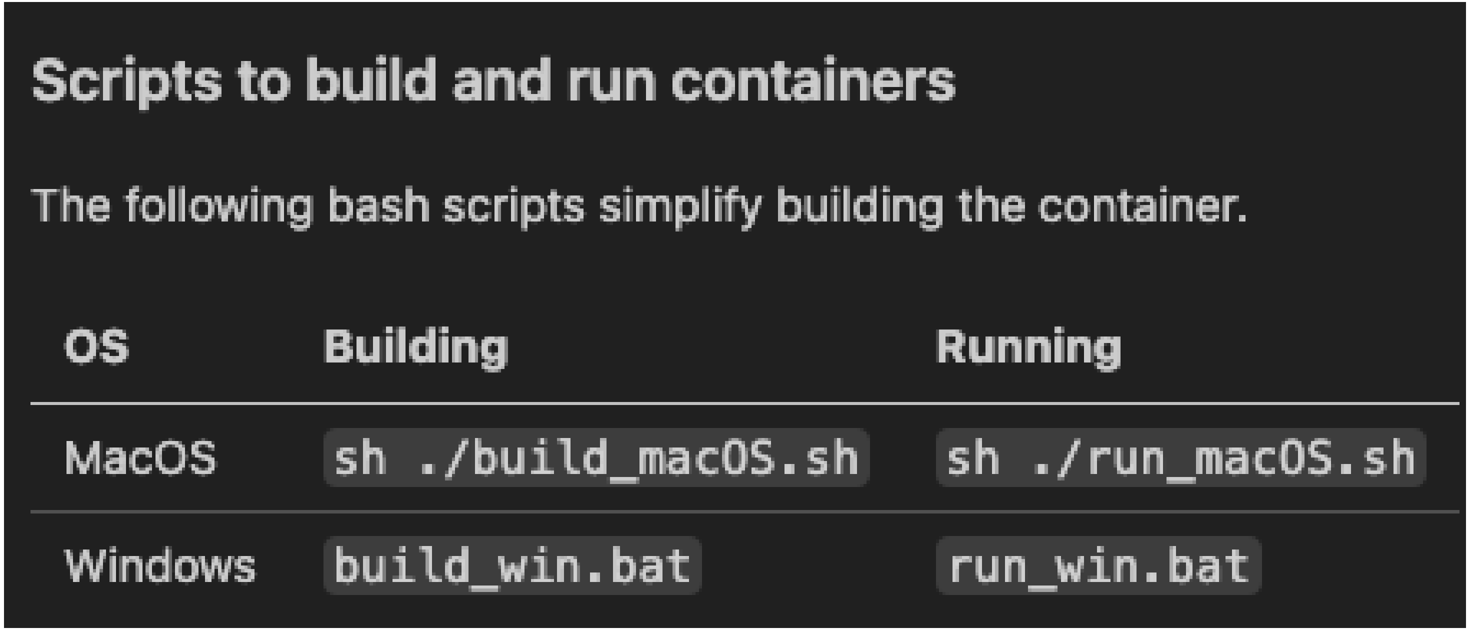
Ways to run SQLite for this demo

- Download and install a local version (Recommended);
  - https://www.sqlite.org/index.html

- Use Docker solution:
  - See classDocs/ for Dockerfile and execution bash scripts

- Use an online tool (Class examples only, not for homework)
  - See sqliteonline at https://sqliteonline.com/

Running SQLite3 from Docker



Scripts to build and run containers

The following bash scripts simplify building the container.

OS	Building	Running
MacOS	`sh ./build_macOS.sh`	`sh ./run_macOS.sh`
Windows	`build_win.bat`	`run_win.bat`

- Create a working directory to contain your working notes and working database file and keep your docker scripts there.
- The directory where you run the Docker virtual machine is the root/ directory. Make sure you can still find your local files wherever when you use Docker

SQL

- Pronounced “ess-que-el” stands for Structured Query Language.
- Used to communicate with a database.
- According to ANSI (American National Standards Institute), it is the standard language for relational database management systems.
- The standard computer language for relational database management and data manipulation.
    - Used to query, insert, update and modify data

SQLite3 - A practical open-source database

Command

- $sqlite3

You should see this, or similar:

```
SQLite version 3.43.2 2023-10-10 13:08:14
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>
```

Create database called dept.sqlite3

- `sqlite3 dept.sqlite3`

Save code and data, and exit using .exit from the sqlite3.

Data and its Schema
We need to tell SQLite3 where to contain the data

Data

Only three columns in our base:

1. ID: up to four chars in size
2. Dept: up to four chars
3. RoomNum: up to 3 chars

Plenty of space for as many rows as we want:

- 1 Limited by memory

Make a General Table

Remove the table to create it with new attributes

- DROP TABLE IF EXISTS tablename;

Code

```
CREATE TABLE table_name (
        column1 datatype,
        column2 datatype,
        column3 datatype
        …
);
```

This data structure allocates the memory space for the database to keep data that is assigned to this table.

Schema - Create a table for the DB

Create database called dept.sqlite3

- sqlite3 dept.sqlite3

```
DROP TABLE IF EXISTS department;

CREATE TABLE department(
          ID VARCHAR,
          Dept VARCHAR,
          RoomNum VARCHAR);
```

- We create a table (a memory space) called department to contain our data
- Note: the VARCHAR attribute is a universal attribute type

After table is created
Add the data

SQLite: What tables are in the DB?

```
sqlite> .tables
department
```

SQLite: What are the attributes and tables of the DB schema?

```
sqlite> .schema
CREATE TABLE department(
ID varchar,
Dept varchar,
RoomNum varchar);
```

Insert some data as a tuple:

```
INSERT INTO department VALUES (
                ”HZ",
                "CS",
                ”114" );
```

After table is created
Add the data

Query everything in the table, department

```
sqlite> select * from department;
HZ|CS|114
```

Query ID and Dept in the table, department

```
sqlite> select ID, Dept from department;
HZ|CS
```

Exit and save your database

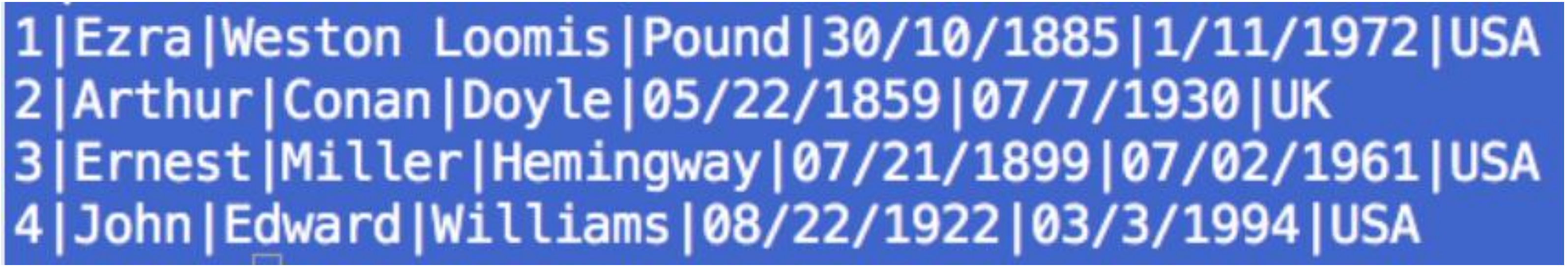
```
.exit
```

Load your database and run same queries again!

```
sqlite3 dept.sqlite3
```

- Can you add and populate a new database?
- Can you populate your base by adding more data?
- Can you also check that the data was correctly added?

Let’s Make another Persistent Database!



```
1|Ezra|Weston Loomis|Pound|30/10/1885|1/11/1972|USA
2|Arthur|Conan|Doyle|05/22/1859|07/7/1930|UK
3|Ernest|Miller|Hemingway|07/21/1899|07/02/1961|USA
4|John|Edward|Williams|08/22/1922|03/3/1994|USA
```

Attributes

- ID
- first name
- middle name
- last name
- birth date
- death date
- country of origin

Create the file!

The terminal command to open a new database

- sqlite3 writers.sqlite3

```
hangzhao@Mac week03 % sqlite3 writers.sqlite3
SQLite version 3.43.2 2023-10-10 13:08:14
Enter ".help" for usage hints.
sqlite>
```

Create the Space

```
CREATE TABLE Writers (
        id INTEGER NOT NULL PRIMARY KEY,
        first_name VARCHAR NOT NULL,
        middle_name VARCHAR,
        last_name VARCHAR NOT NULL,
        birth_date VARCHAR NOT NULL,
        death_date VARCHAR,
        country_of_origin VARCHAR NOT NULL );
```

- Note: NOT NULL ensures that this field is not left blank when populating

Add Data for writers Table

Insert Commands

```
INSERT INTO Writers VALUES(1, ’Ezra’, ’Weston Loomis’, ’Pound’, ’30/10/1885’, ’1/11/1972’, ’USA’);
INSERT INTO Writers VALUES(2, ’Arthur’, ’Conan’, ’Doyle’, ’05/22/1859’, ’07/7/1930’, ’UK’);
INSERT INTO Writers VALUES(3, ’Ernest’, ’Miller’, ’Hemingway’, ’07/21/1899’, ’07/02/1961’, ’USA’);
INSERT INTO Writers VALUES(4, ’John’, ’Edward’, ’Williams’, ’08/22/1922’, ’03/3/1994’, ’USA’);
```

Tables and Schema

- What is the schema (i.e., the arrangement of data) of your database?
  - Type in “.schema” and see!
- What are the tables of your database?
  - Type in “.tables” and see!

Conditional Queries
Adding conditional clauses to queries

Queries to play with using conditional clauses

```
select * from Writers where country_of_origin == "UK";
select * from Writers where country_of_origin == "USA";
select * from Writers where birth_date == "08/22/1922";
select * from Writers where first_name == "Arthur";
```

Save your DB and exit with following command:

```
.exit
```

- Can you populate your base by adding more data?
- Can you also check that the data was correctly stored in the table?
- Can you run queries to access attributes?