



# Introduction to Database Systems: CS312 A Small Database System

Oliver Bonham-Carter

01 March 2022

# All types of data!

## Data Models

Types of  
bases

Ready  
SQLite3

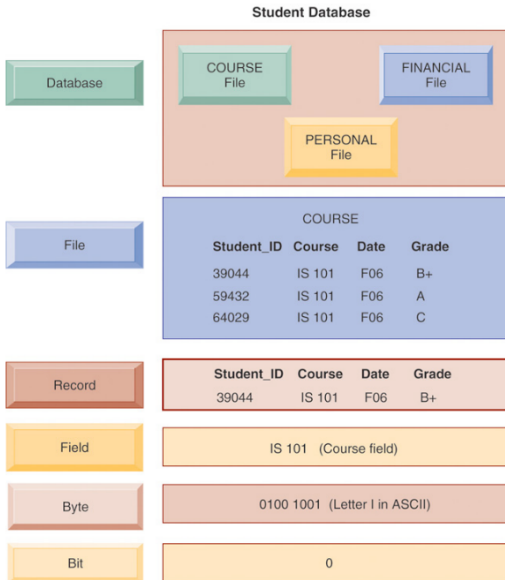
Our First DB

Data to add

Create table

Schema

Consider  
this...



# A database, simply stated

## Data Models

### Types of bases

#### Relational Models

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

ID	Dept	RoomNum
JJ	CS	105
OBC	CS	104
AM	CS	106
GK	CS	108
PL	CS	110
DW	CS	112

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

# A database, not-so-simply stated

Data Models

Types of  
bases

Relational Models

Ready  
SQLite3

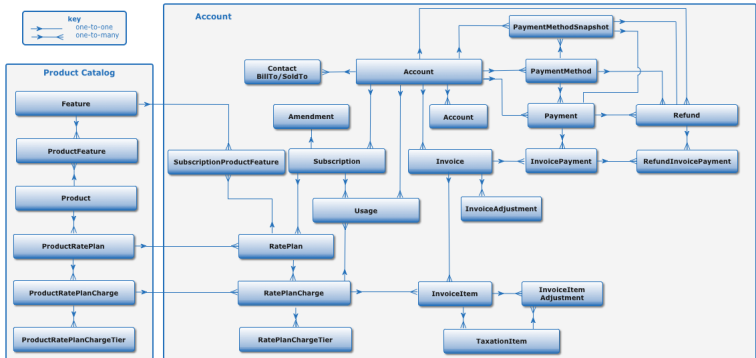
Our First DB

Data to add

Create table

Schema

Consider  
this...



- 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

Data Models

Types of  
bases

Relational Models

Ready  
SQLite3

Our First DB

Data to add

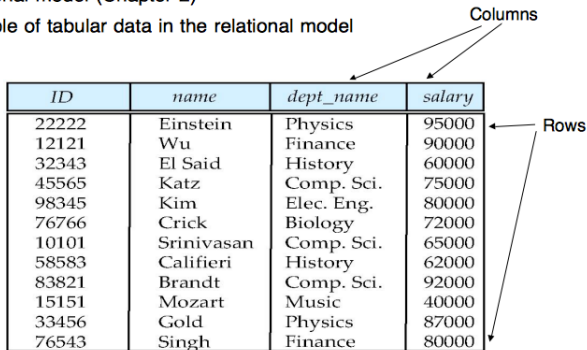
Create table

Schema

Consider  
this...

## Relational model (Chapter 2)

Example of tabular data in the relational model



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

(a) The *instructor* table

- 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

Two tables containing *specific* types of data

Data Models

Types of  
bases

Relational Models

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

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

(a) The *instructor* table

<i>dept_name</i>	<i>building</i>	<i>budget</i>
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

# Specific information for each table

Data Models

Types of  
bases

Relational Models

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

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).

# We've got it!

Data Models

Types of  
bases

Relational Models

Ready  
SQLite3

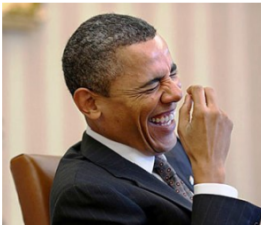
Our First DB

Data to add

Create table

Schema

Consider  
this...



## Great!!

- Let's build a small-sized database using SQLite3!!



# Running an SQLite client

Data Models

Types of  
bases

Ready  
SQLite3

Docker

Our First DB

Data to add

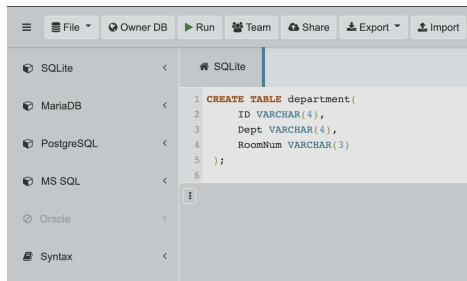
Create table

Schema

Consider  
this...

## Ways to run SQLite for this demo

- Download and install a local version;
  - See supplemental slides about *Tools*
- Use an online tool (shown below)
  - See *sqliteonline* at <https://sqliteonline.com/>
- Use Docker solution:
  - See *classDocs/* for Dockerfile and execution bash scripts



# Running SQLite3 from Docker

Data Models

Types of  
bases

Ready  
SQLite3

Docker

Our First DB

Data to add

Create table

Schema

Consider  
this...

## Shortcut to running and building a working container

The following bash scripts simplify building the container.

OS	Building	Running
MacOS	<code>./build_macOS.sh</code>	<code>./run_macOS.sh</code>
Linux	<code>./build_linux.sh</code>	<code>./run_linux.sh</code>
Windows	<code>build_win.bat</code>	<code>run_win.bat</code>

- Create a working directory to contain your working notes and working database file.
- If you cannot find an easy way to access your Docker builder and initiation scripts (`docker-databases/`), then copy over the files into your working directory. Initializing your container there makes that directory the root once inside the container.
- If you installed SQLite3 locally, then you should be able to execute it anywhere in your computer.

Data Models

Types of  
bases

Ready  
SQLite3

Docker

Our First DB

Data to add

Create table

Schema

Consider  
this...



- 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

## Command

```
$sqlite3
```

You should see this, or similar:

```
SQLite version 3.19.3 2017-06-27 16:48:08
```

```
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*.

# We are going to build this database

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

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

- Our database will contain this same ordering of data

# Data and its Schema

We need to tell SQLite3 where to contain the data

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

## 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

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

## Pseudo 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

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

Create database called *dept.sqlite3*

```
$sqlite3 dept.sqlite3
```

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

- We create a table called *department* to contain our data
- In fact, we have created a *memory space* for this task
- Note: the VARCHAR attribute
  - SQLite3 does not impose any length restrictions on the length of strings, BLOBs or numeric values.



# After table is created

Add the data

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...

Check that the table has been created

```
sqlite> .tables  
department
```

Insert some data as a *tuple*

```
INSERT INTO department VALUES (  
    "OBC",  
    "CS",  
    "104" );
```

Query everything in the table, department

```
sqlite> select * from department;  
OBC|CS|104
```

Exit and save your database

```
.exit
```

# Consider this...

Data Models

Types of  
bases

Ready  
SQLite3

Our First DB

Data to add

Create table

Schema

Consider  
this...



# THINK

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?