DB Class Notes

Followup for CSC330 Database Theory & Applications Spring 2022

Table of Contents

README

Instead of bugging you with emails, I opt to summarize my course observations regarding content, process, in this file. These often contain additional links, articles, and musings.

I usually update it after each class - it also contains the homework (if any). The first point of call for any questions should be the FAQ. There are two FAQs - a general one (for all my courses), and a FAQ for CSC 330.

You find the whiteboard photos here in GDrive.

The companion file to this file, with the agenda and much of the course content, is the agenda.org file.

Intro and GitHub - w1s1 (01/11/22)

Homework (by Thursday 13-Jan)

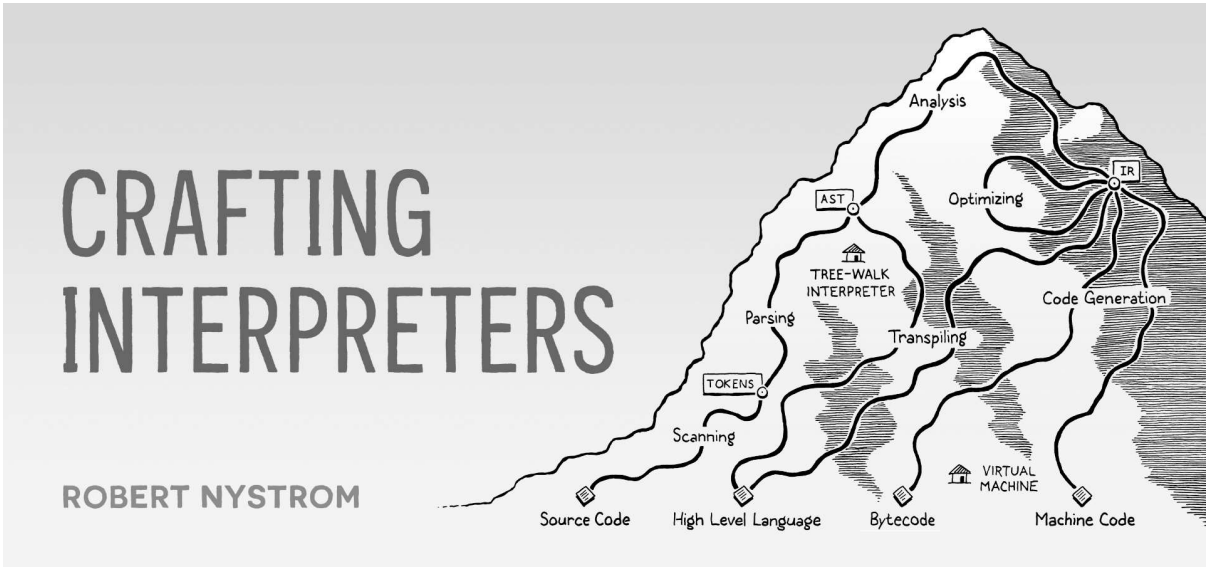
- Register with GitHub (2 min)
- Complete the GitHub Hello World exercise (20 min)
- ~~Give me your GitHub user name (email - with course ID - or Thursday in class) - so that I can add you to the repo~~ - no need, course is public now
- Submit an issue to the course repo confirming that you completed the exercise.

Note: if you already did this as part of another course, you don't have to do it again but you need to let me know in your email, which of my courses you attend so that I can add you to one or more GitHub repos.

Stuff

- Agenda - we covered all of it (and more) - agenda is available in GitHub only.

- Books: "Crafting Interpreters" by Nystrom (2021) is a fun book on creating a small interpreted programming language. This would be an extra cool (Honors) project in a course on programming languages!



- Pyret (thank you, Molly!) - check it out.

- Showed GitHub, DataCamp teams and assignments, and GNU Emacs. We talked about structured vs. unstructured data, and Torvald's Git version control program: which follows a key-value data model - all data (committed differences) are stored in tree-like structures and indexed by hash codes. See here for more (Spajic, 2018).

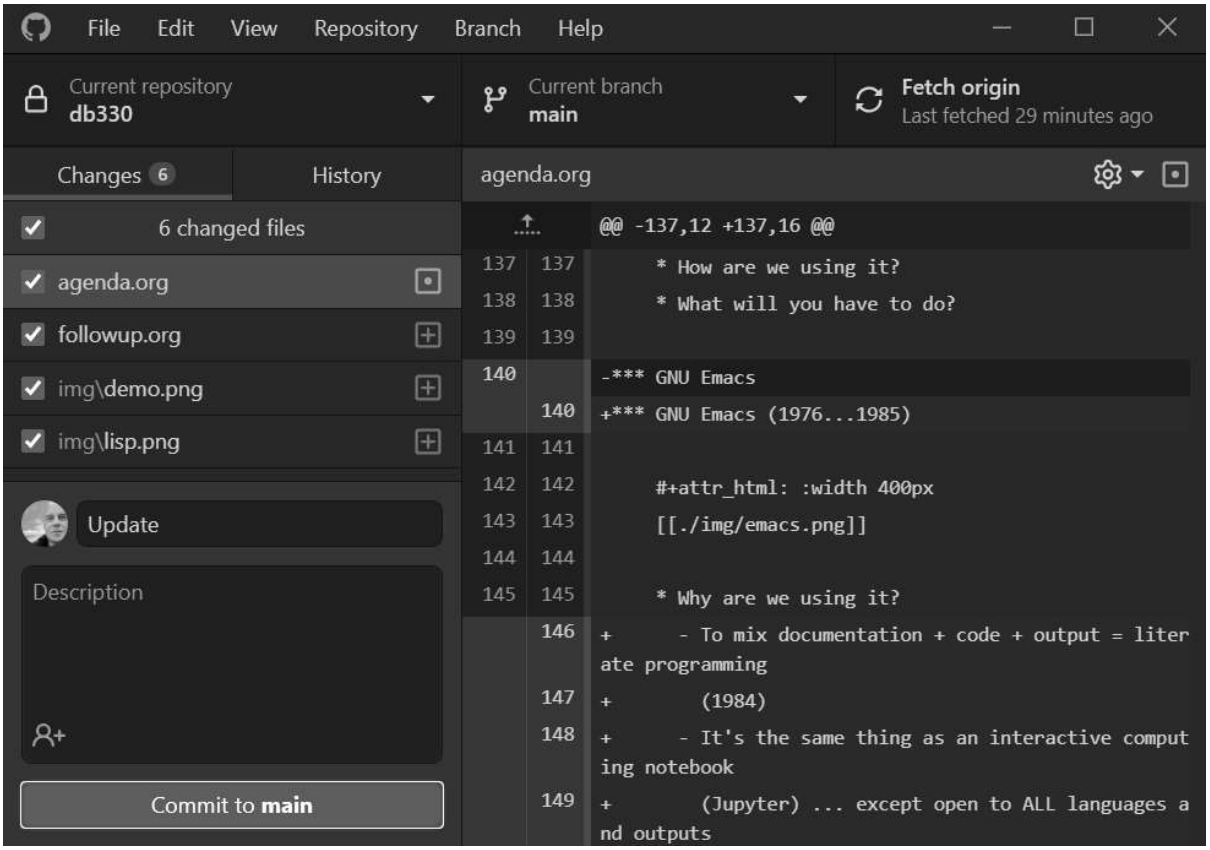


Figure 2: GitHub desktop application

- The DataCamp platform supports the three most important data science languages (in order of importance in the real world): SQL, R, Python. You can do data science with the shell (bash), or even with Java, C, or any other language of course. The season's last assignment is a project where you can see all that you learnt about SQL at work: an anlysis of international debt statistics. You do not need to work on the assignments in any particular order - as long as you don't miss the weekly deadlines. A weekly assignment should take 20-30 minutes of your time.

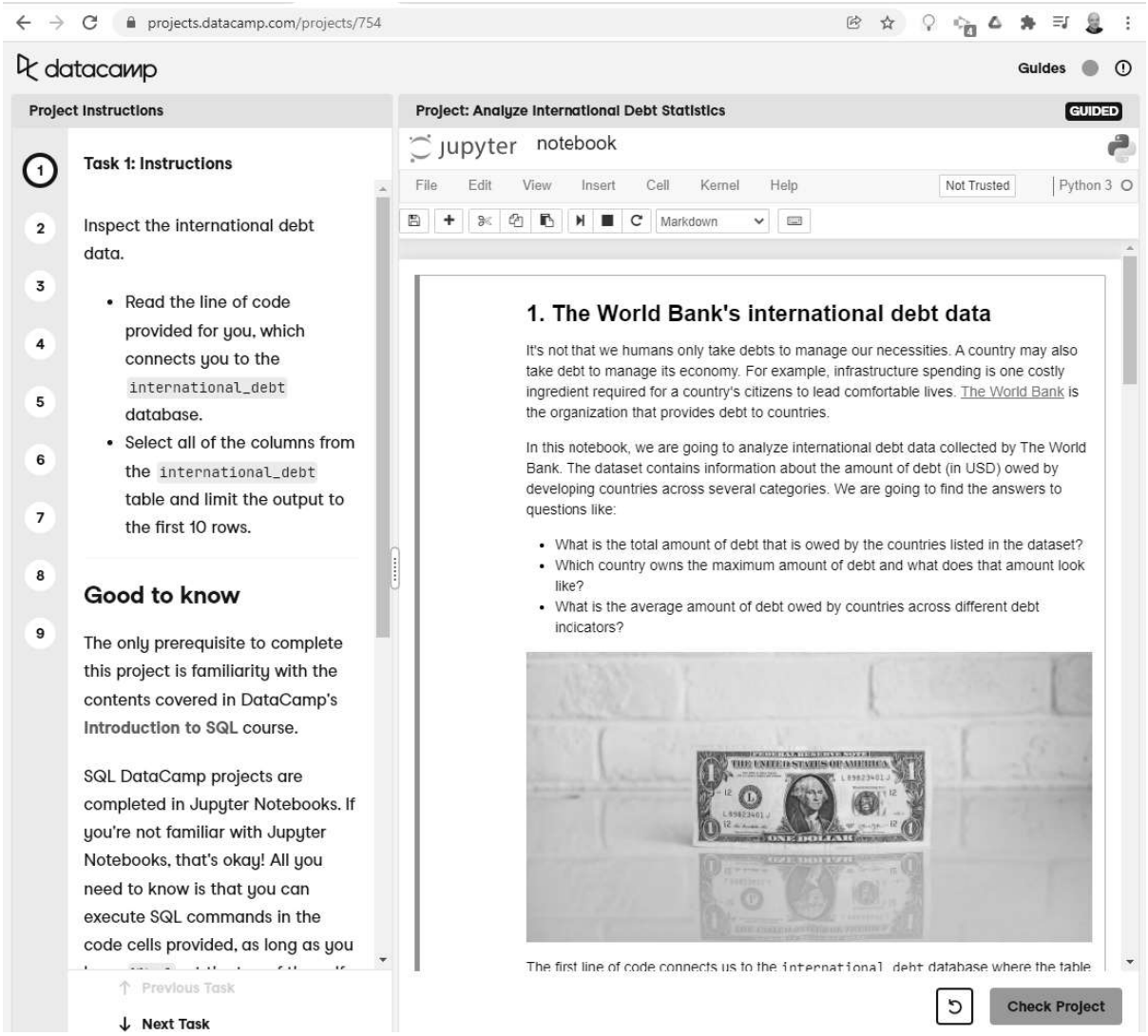
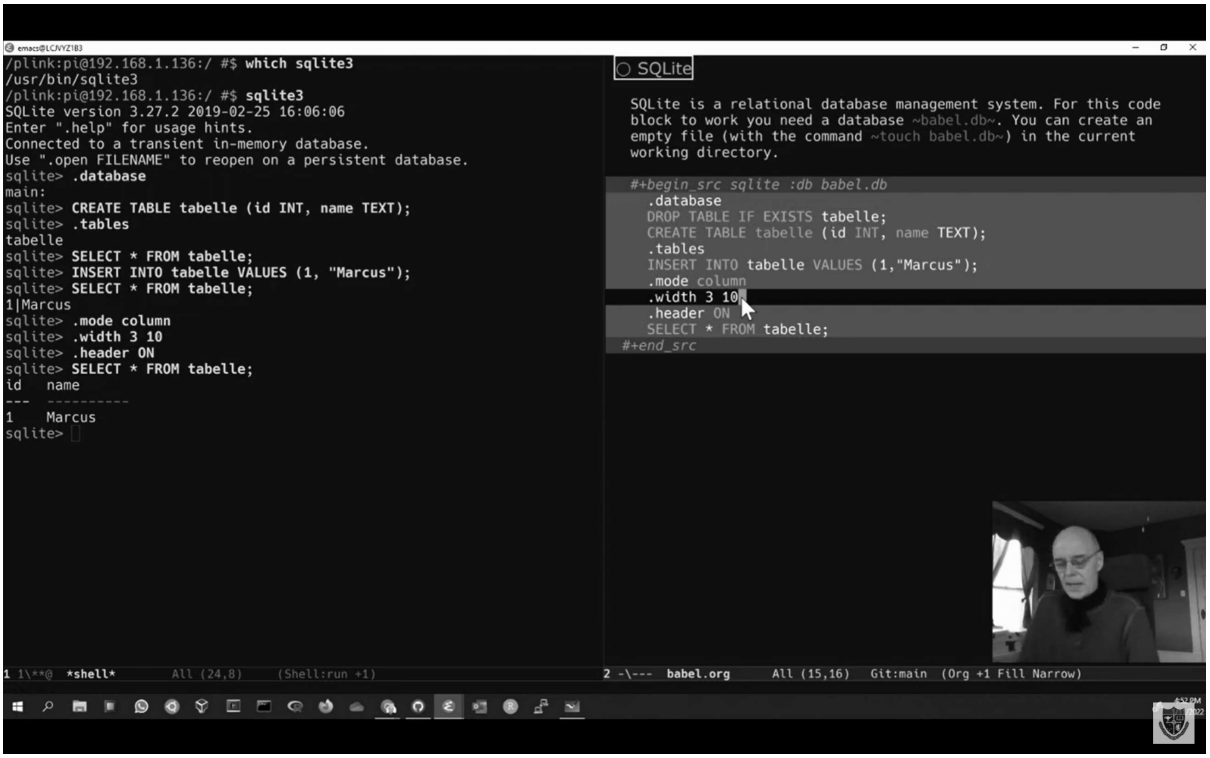


Figure 3: DataCamp project - Analyze Intern. Debt Stats

- Recorded a screencast for the missing 10 minutes at the end (network outage in Batesville!) - see on YouTube here (Birkenkrahe, 2022). This is a demonstration of the differences of interactive shell scripting vs. interactive notebooks, using SQLite as a sample language. You will get an assignment to create a literate programming file next week. Don't be put off by the level of detail of this demo - let it all just wash over you for now - you'll learn this and much more in this course!



In the screencast I talk about missing syntax highlighting in the shell buffer on the left: "I could add it if I wanted to." This is true - everything is Emacs is customizable, and I spent years, literally, doing this. Here is the documentation specifically on customizing the shell, if you're curious. The language to do this is Emacs-Lisp, a Lisp dialect. I mentioned that Lisp was the first and, for a long time, dominant AI language (Valencia, 2017). It's a great language to learn, and GNU Emacs is the ticket if you feel like it.

```
(defun make-perceptron (n m &optional (g  #'(lambda (i) (step-function 0 i)))
    &aux (l nil))

    (dotimes (i m (list l))
        (push (make-unit :parents (iota (1+ n))
            :children nil
            :weights (random-weights (1+ n) -0.5 +0.5)
            :g g)
        l)))
```

Figure 5: Common Lisp code to create an n-inputs m-units one layer perceptron. Source: AIMA.

DataCamp, History of DB, MooCall - w1s2 (01/13/22)

DataCamp membership

- DataCamp: You should all be in your courses now.
    - Your assignments are on one page but you'll be notified via schoology as soon as an assignment is due

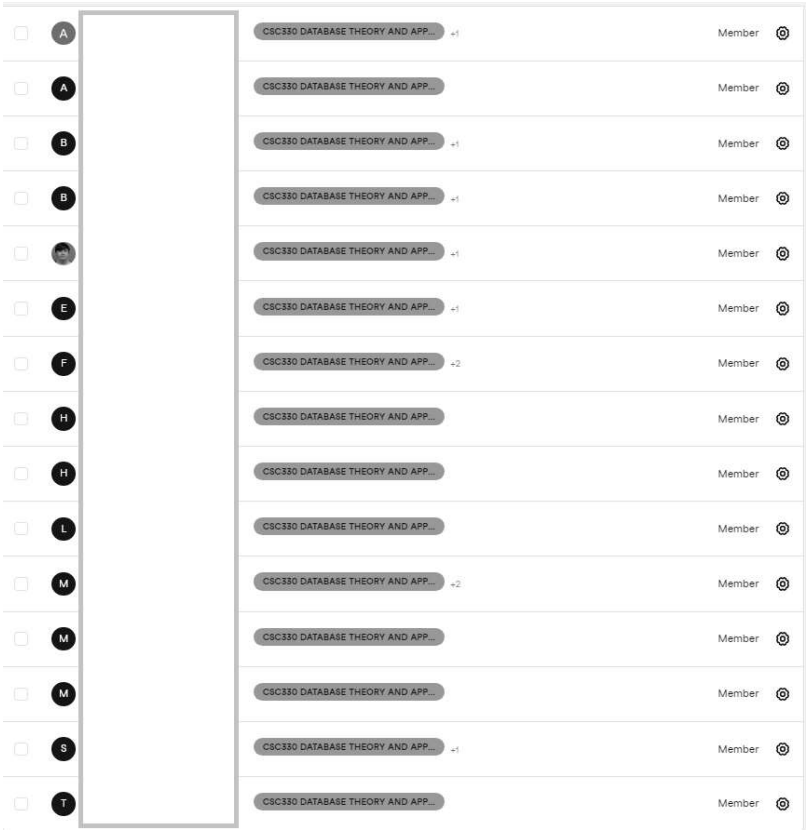


Figure 6: DataCamp DB course member list

History of databases

- YouTube video link

Figure 7: Which of these founders has nothing to do with databases?

- DB are an old human interest (information is always gathered)

- DB development happened in close "combat" with companies and operating systems (market and a hardware aspect to it)
- Dominant language is SQL - in connection with relational DB
- Codd's 12 rules = everything you need to know about relational DB design (these rules scale extremely well)
- There are MANY different types of DBMS
- We'll be installing SQLite, too, for local SQL experimentation

DBMS IoT example application (MooCall)

- Website for this Irish app

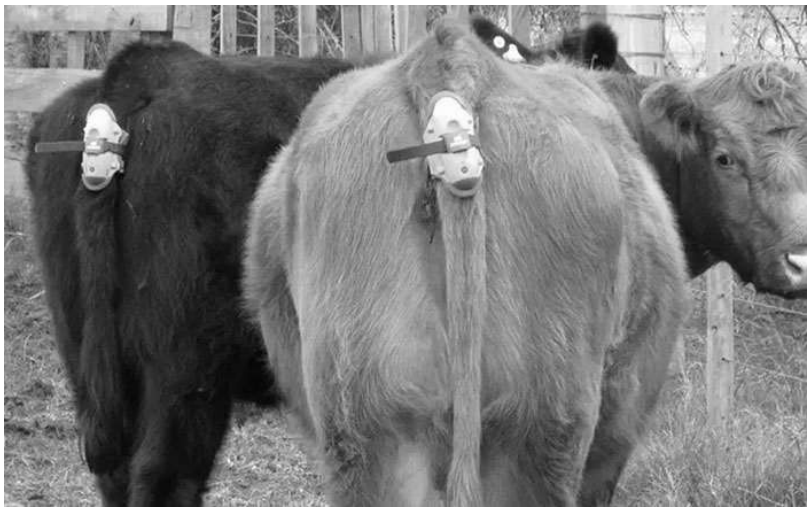


Figure 8: Two cows with MooCall sensors attached.

- Sensor applications are small and look trivial, but they're highly security relevant (Cernobyl disaster in USSR, 1986). See The Logic of Failure (Dorner, 1997).

- Big Data ("starts" modern data science) = 3Vs = Volume + Velocity + Variety [5V definitions add "value", "veracity"]
- IoT = "webservice"-enabled, cloud-networked, fast, big data applications
- MooCall DB networking:
    1. cow data are generated and pre-processed locally (e.g. cow ID, temperature, motion etc.) = "edge computing" example
    2. cow data are processed globally (in the cloud) to generate user signal ("this cow's calving!")
    3. Signal is transmitted to the user = farmer for potential action.
- DB system = DBMS + DB - the DBMS has a lot of fancy stuff on board: performance optimizer, shell, API
- BC (Before Codd) = file system-based data management, and AC (After Codd) = DB management system (DBMS)-based data management.

GNU Emacs installation

- Install vanilla[1] GNU Emacs or a modified Emacs (for statistics processing with R and other packages - ESS)
- Vanilla GNU Emacs v27 Windows installation: you need the `emacs-27.1-x86_64-installer.exe` from this page.

- MacOS: get the modified version if you like (easiest), or the binary using the command line terminal as shown here, using the `Homebrew` package manager.

Index of /emacs/windows/emacs-27

Name	Last modified	Size	Description
Parent Directory		-	
emacs-27-deps-mingw-w64-src.zip	2020-08-25 12:28	107M	
emacs-27-deps-mingw-w64-src.zip.sig	2020-08-25 12:28	833	
emacs-27.1-i686-installer.exe	2020-08-24 17:47	59M	
emacs-27.1-i686-installer.exe.sig	2020-08-24 17:47	833	
emacs-27.1-i686-no-deps.zip	2020-08-25 12:20	47M	
emacs-27.1-i686-no-deps.zip.sig	2020-08-25 12:20	833	
emacs-27.1-i686.zip	2020-08-25 12:22	102M	
emacs-27.1-i686.zip.sig	2020-08-25 12:22	833	
emacs-27.1-x86_64-installer.exe	2020-08-25 12:23	59M	
emacs-27.1-x86_64-installer.exe.sig	2020-08-25 12:23	833	
emacs-27.1-x86_64-no-deps.zip	2020-08-25 12:24	47M	
emacs-27.1-x86_64-no-deps.zip.sig	2020-08-25 12:24	833	
emacs-27.1-x86_64.zip	2020-08-25 12:26	103M	
emacs-27.1-x86_64.zip.sig	2020-08-25 12:26	833	
emacs-27.2-i686-installer.exe	2021-03-31 12:45	60M	
emacs-27.2-i686-installer.exe.sig	2021-03-31 12:45	833	
emacs-27.2-i686-no-deps.zip	2021-03-31 12:45	47M	
emacs-27.2-i686-no-deps.zip.sig	2021-03-31 12:45	833	
emacs-27.2-i686.zip	2021-03-31 12:45	102M	
emacs-27.2-i686.zip.sig	2021-03-31 12:45	833	
emacs-27.2-x86_64-installer.exe	2021-03-31 12:45	60M	
emacs-27.2-x86_64-installer.exe.sig	2021-03-31 12:45	833	
emacs-27.2-x86_64-no-deps.zip	2021-03-31 12:46	47M	
emacs-27.2-x86_64-no-deps.zip.sig	2021-03-31 12:46	833	
emacs-27.2-x86_64.zip	2021-03-31 12:46	103M	
emacs-27.2-x86_64.zip.sig	2021-03-31 12:46	833	

Figure 10: GNU Emacs v27 download online repo

- Demo: keyboard macros. I showed how to define a keyboard macro (`C-x (` [key sequence] `C-x )`) and how to apply it repeatedly (`C-u [times] C-x e`). Here's the documentation.

DB elements, GNU Emacs - w2s3 (01/18/22)

Quiz 1

THE QUIZ IS ON … SCHOOLOGY 1 PM - 1.15 PM

Feedback/discussion

FOLLOWED by brief FEEDBACK:

- We'll do one of these per week (I hope)
- Any content questions?
- Too much time? Too little?
- Questions too hard? Too easy?
- A subset of these questions will become the final exam
- After playing the quiz in class you can play it unlimited times
- Quizzes are now **ungraded** (final exam is now 30% of final grade)

Review: file vs database approach

DICT	
TABLE_NAME	COMMENTS
USER_RESOURCE_LIMITS	Display Resouce limit to the User
USER_PASSWORD_LIMITS	Displays password limits to the User
USER_CATALOG	All tables, views,synonyms, Sequences owned by the users
ALL_CATALOG	All tables, views,synonyms, Sequences accessible to the users
USER_CLUSTURES	Description of user's own Clustures
ALL_CLUSTERS	Description of clustures accessible to other users
DBA_TABLES	Description of all relational tables in the database
DBA_ALL_TABLES	Description of all object and relational tables in the database
USER_TABLES	Description of the user's own relational tables
ALL_TABLES	Description of relational tables accessible to the user
ALL_INDEXES	Descriptions of indexes on tables accessible to the user
DICT	Synonym for DICTIONARY
COLUMN_PRIVILEGES	Grants on columns for which the user is the grantor, grantee, owner

Figure 11: database dictionary example (tutorialcup.com)

Captain's Log[2]

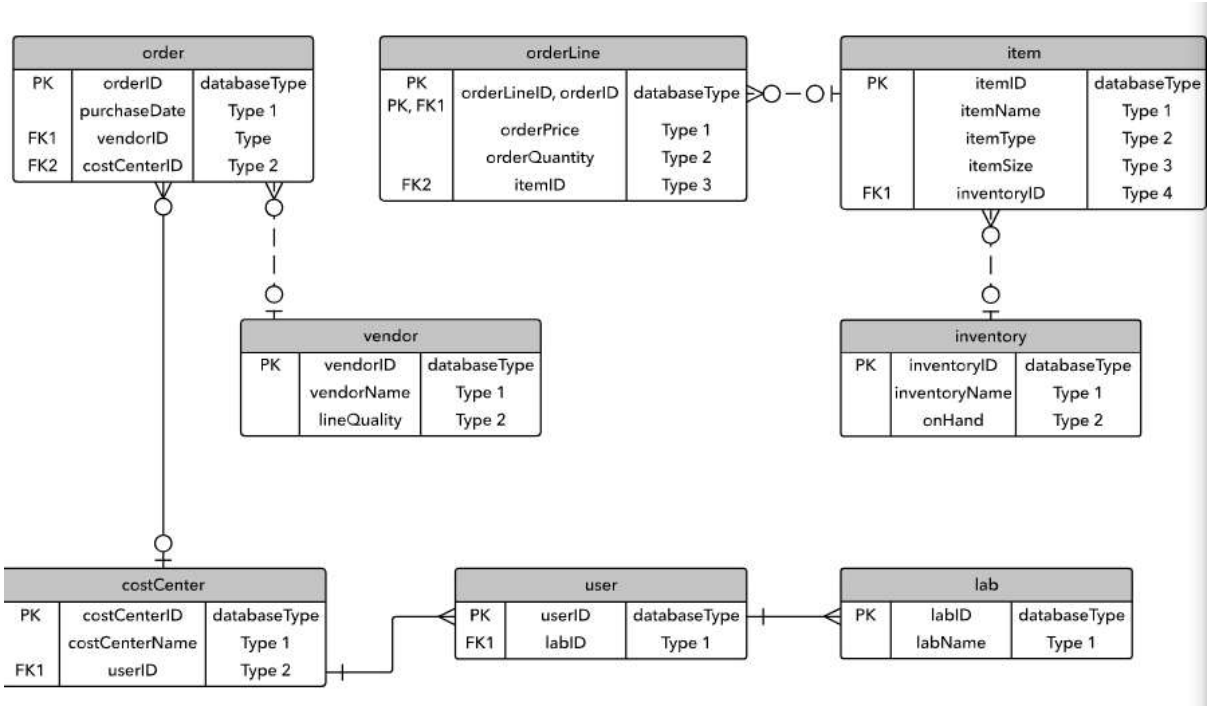


Figure 12: Entity Relationship Diagram (Source: Lucidchart).

- strong vs. soft/loose coupling is an important design issue (it relates directly to the resilience of the designed system).
- **meta data** are data about data, used to control, manage other data, and processes
- Example: Exif (Exchangeable Image File) data are an example of common meta data associated with image files stored alongside digital pictures.
- SQL (Structured Query Language): dominant language for structured, table-based (aka relational) databases
  - DQL = Data Query Language (e.g. selecting data)
  - DDL = Data Definition Language (e.g. creating tables)
  - DML = Data Manipulation Language (e.g. input/output of data)
  - DCL = Data Control Language (to alter meta data, e.g. rights)
- Learning anything is best as a "variation on a theme" (as in music)
- MooCall: what do we want to store (cow temperature F, in C)
- State-based view of computing (von Neumann architecture) = dominant paradigm (automata, Turing machines…)

GNU Emacs tour

- Ctrl-h Ctrl-a RET : Startup screen
- Emacs written in C and Lisp (Emacs Lisp)
- Emacs is an IDE - we'll use it for SQL, SQLite, and bash
- Emacs contains a bunch of apps (e.g. file explorer)
- Try the Emacs onboard tutorial (CTRL-h t)
- What Emacs can do:
  - Extension and full customization (with Emacs Lisp)
  - Writing in many different human/programming/markup languages (with major and minor modes)
  - IDE work (compile, run, test programs) - gdb integration
  - Compare and highlight file differences (with ediff)
  - Manage files (with dired)
  - Read mail, news, RSS feeds (gnus)
  - You can use it as an IRC reader (#batesville@irc.freenode.net)
  - Play games (examples)

> Whenever you decide to start using Emacs, you should take the Emacs tutorial. It's an interactive hands-on which will familiarize you with many things, including:
>
> - Starting and exiting Emacs
> - Basic text movement and editing commands
> - Opening and saving files
> - Emacs concepts: windows, frames, files, and buffers
> - Invoking commands with keybindings and with M-x
> - To run the tutorial, start Emacs and type C-h t, that is, Ctrl-h followed by t.

SQLite installation - w2s4 (01/20/22)

Captain's Log

- GNU Emacs reference car ("cheat sheet") on GitHub (PDF)
- We looked at different reasons to use the Emacs editor/IDE
- For installation of the software we need (Emacs, SQLite), which is already done on the PCs in the computer lab, check install.org in GitHub

- To get better at moving through Emacs buffers, manage files etc., complete the GNU Emacs onboard tutorial (open it in Emacs with `C-h t`)
- We'll go through a more systematic training session (including Org-mode) next week.

Cloud computing intro - w3s5 (01/25/22)

Cloud computing - "the old curmudgeon's view"

- Spending on cloud services in 2020: $bn 266 - projected to $bn 308 in 2021 (DataCamp). However (IDC, 2022)

> For the full year 2021, IDC forecasts cloud infrastructure spending to grow 8.3% compared to 2020 to $71.8 billion, while non-cloud infrastructure is expected to grow 1.9% to $58.4 billion after two years of declines. Shared cloud infrastructure is expected to grow by 7.2% year over year to $49.7 billion for the full year. Spending on dedicated cloud infrastructure is expected to grow 10.7% to $22.2 billion for the full year.

Figure 13: Cloud service spending data (IDC, 2022)

- What's the GDP of Finland and Vietnam?
- On-premise vs cloud discussion - major business issue - why?
- Main messages: cloud is huge, good for access, scaling is an issue
- Amazon Web Services is highlighted (why?) What about other providers? How do they differ? What kind of market is this?

- What do you think of the AWS products page from a logical point of view?

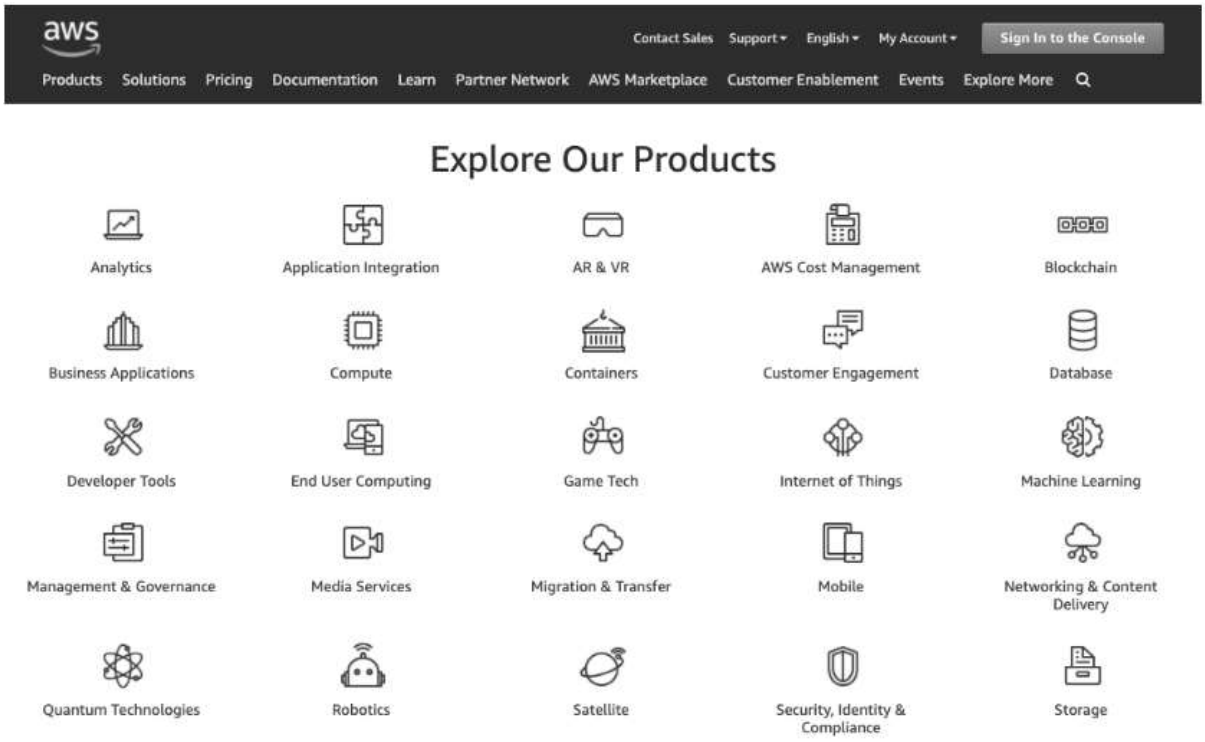


Figure 14: aws products and services (DataCamp, 2020).

- Virtualization = your OS over the internet
- Vertical (server power) vs. horizontal scaling (server number)
- Data centers - cp. "Inside Google's Data Centers" (H5P/YouTube)

- How are Lyon's data served - what about the safekeeping?

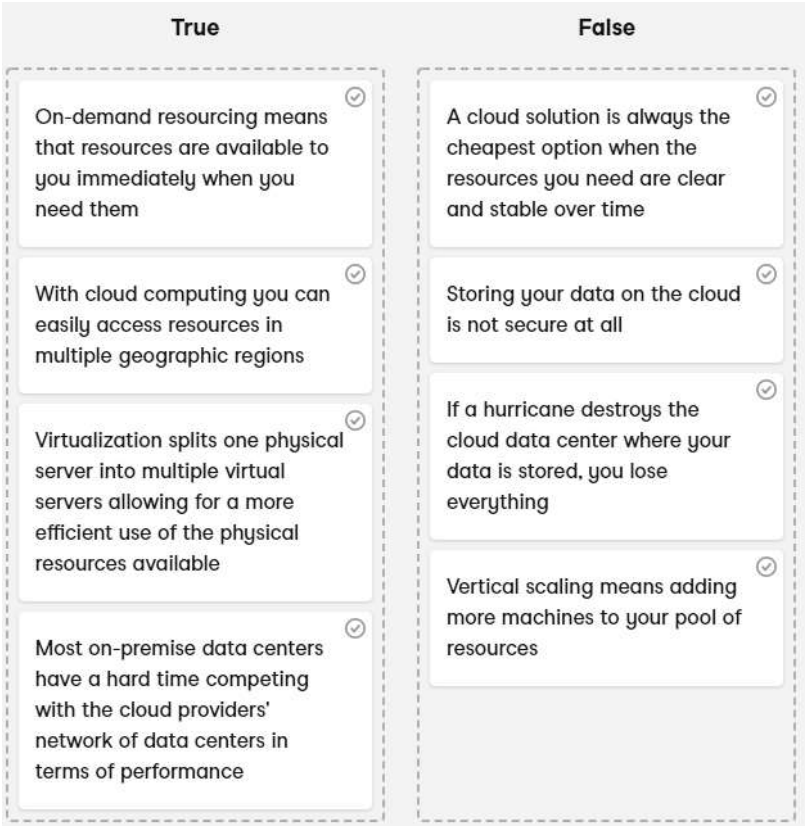


Figure 15: On-demand cloud vs. on-premise data centers

- IaaS/PaaS/SaaS: Socialist propaganda or brave new world of shared services? What're some key conditions for this to be beautiful?

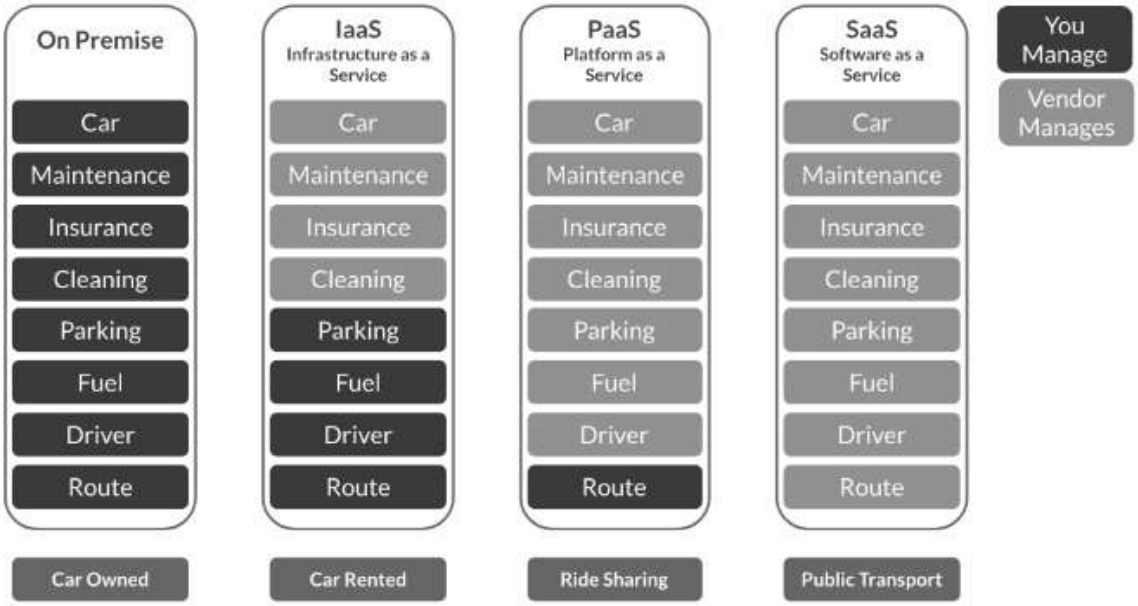


Figure 16: Cloud service models - car analogy (DataCamp, 2020)

- IT services

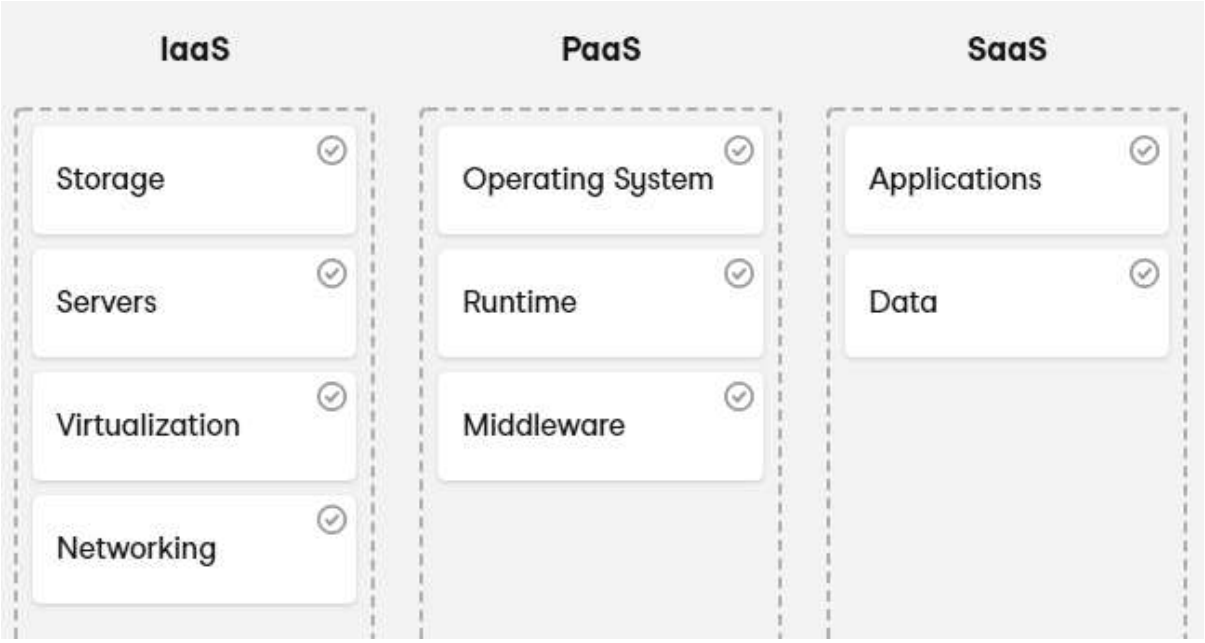


Figure 17: IaaS vs. PaaS vs. SaaS (DataCamp, 2020)

- Use cases (UML anyone?)

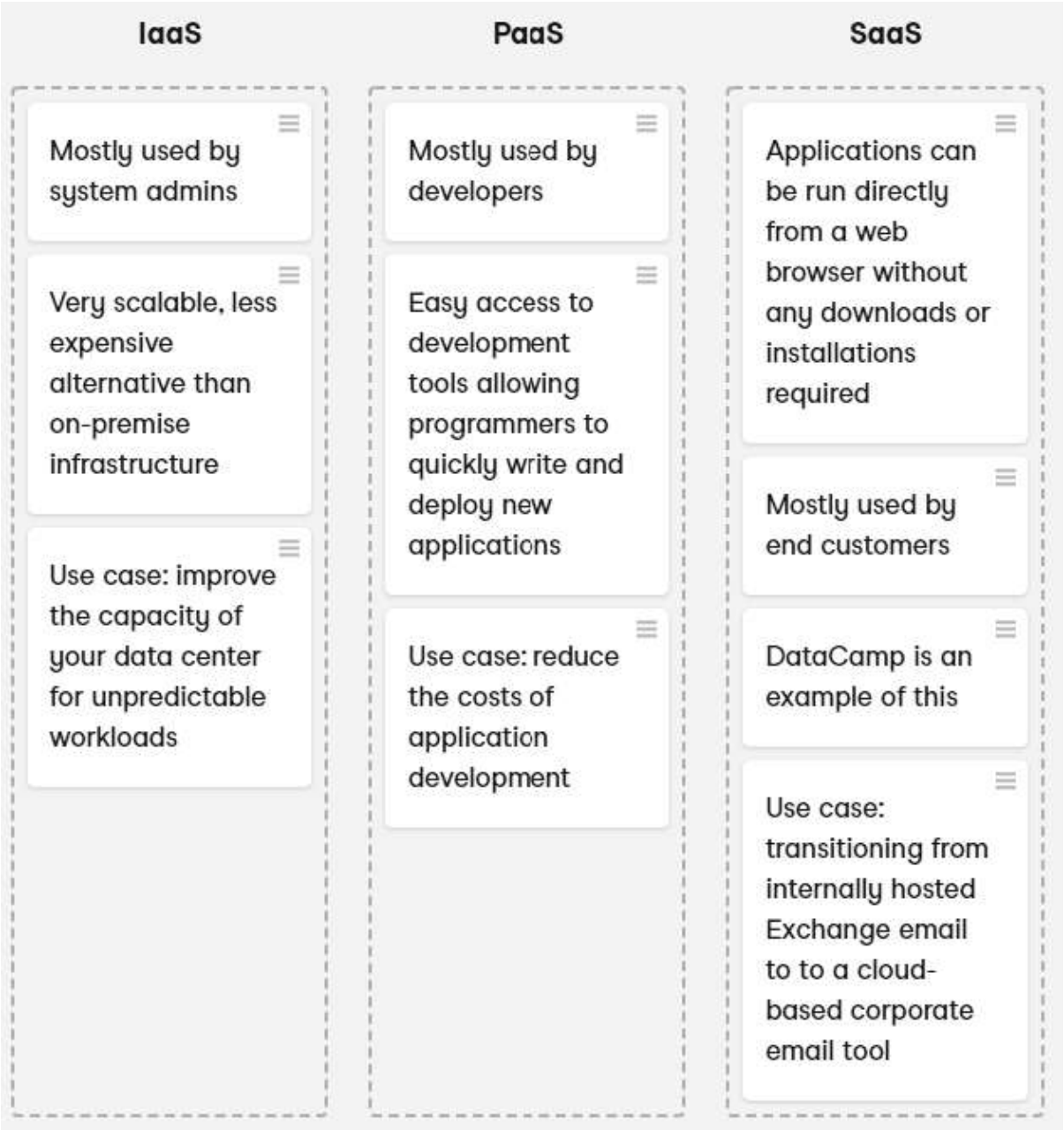


Figure 18: IaaS vs. PaaS vs. SaaS (DataCamp, 2020)

- The cloud pyramid - "More complexity means less control"

Figure 19: Control vs. Abstraction (DataCamp, 2020)

Captain's Log Star Date 99667.82

- Use case diagrams are part of the Unified Modeling Language (UML). Here is a good, short, free online video tutorial (Lucidchart, 2018). UML is a visual language to describe information systems.
- We discussed a bank (more specifically a retail, or consumer bank) as an example. It is not a good use case for cloud computing, because the data are confidential, structured, don't change much, and are small. Banks more often use on-premise relational database management systems (RDBMS) and SQL, than cloud-based systems.
- When we ask (about a table, or a diagram, or a graph, or any scheme), "is this logical?", we mean "is this well ordered, or ordered at all, fit for human understanding." There is a whole method, the so-called "Minto Pyramid Principle" built around this concept of ordering content so that it is fit for human understanding. Here is a short online video tutorial on Minto (Harrison Metal, 2019).
- "Shell inside a shell": when we open the Windows terminal, or Command prompt (`cmd.exe`), and then start a program like `sqlite3`, we operate in a shell inside another shell.
- REPL = Read-Eval-Print-Loop - cloud-based application to learn programming: replit.com.

Cloud deployment - w4s6 (02/01/22)

DataCamp assignment - cloud deployment

- Deployment models: private, public, hybrid, multicloud, community
- Negative personal example (hybrid): RStudio cloud
- Positive personal example (public): Colaboratory + GDrive, DataCamp Workspace
- Nice exercise: build a NextCloud server with Raspberry Pi
- EU GDPR - the true story: a major pain with unclear gains
- Is there a US equivalent of the General Data Privacy Regulation act of the EU?[3]
- "What is personal data?" is an interesting question - why?[4]
- How international is the Internet really? (What is its backbone, where are public data held, and who owns the infrastructure?
- How do "cloud computing roles" relate to "database roles"?
- How can you improve your "cloud computing skills"? What are they? (Examples: IBM@coursera.org, EdX cloud computing courses, Google)

- Overlap with data science: analysis (EDA), engineering (pipeline building and maintenance), and modeling (ML)
- Database apps: Tableau etc. all based on SQL = top skill
- Buzzword (2010-2020): "Digital transformation of the company"

Figure 20: Microsoft Skills Report (2017)

Captain's Log Stardate 99687.04

- If you don't have the Linux program `touch` on Windows, you can use this command to create an empty file in the current directory:

```
$ fsutil file createNew test.db 0
```

- An even simpler way is by opening the RDBM program and create a DB at the same time with the command `sqlite3 test.db` at the prompt.
- NextCloudPi contains documentation to use the cloud computing software `NextCloud` on the Raspberry Pi. Somewhere in there you'll find what it takes to build a backup server with `NextCloud` (which was one of my projects).
- World-Wide Web vs. Internet (BBC, 2019):

  "The World Wide Web Is Not The Internet!"

Cloud providers, SQLite introduction - w5s7 (02/08/22)

DataCamp assignment - Cloud providers and case studies

Figure 21: The benefits of French culture (Source: ila-france.com)

Disclaimer (limitations / bias statement)

- "Customers churn"?
- Commercial examples (no scientific objectivity)
- Case studies with a strong French accent (your reaction?)

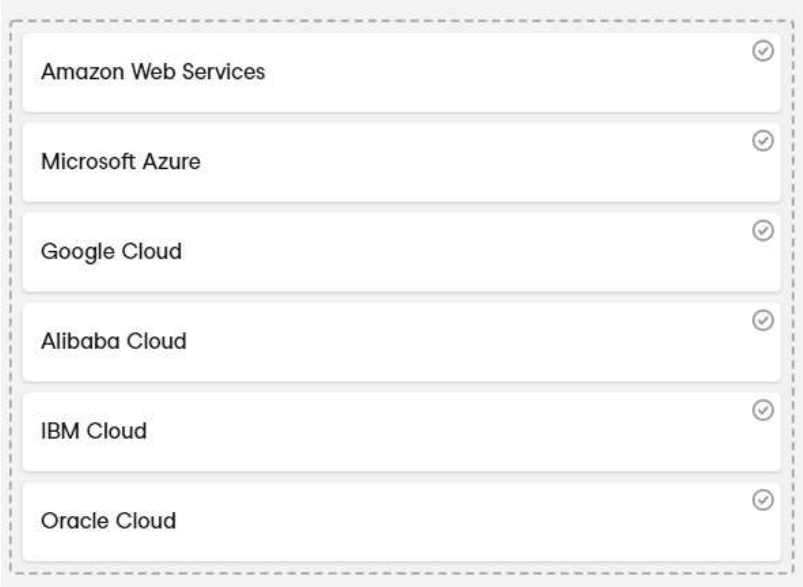


Figure 22: Global cloud service providers (DataCamp, 2019)

AWS - Amazon Web Services

- Q4/2019 - pre-pandemic (changes in 2021?[5])

WHAT	2019	2021
market share	32.4%	51.1%
revenue	$9.8bn	$17.8bn
annual revenue growth	33.2%	40%

- First mover advantage
- Are there jobs with AWS in Arkansas?
- Does case example Nerdwallet (FinTech) still exist?

Microsoft Azure

- Q4/2019 - pre-pandemic (changes in 2021?[5])

WHAT	2019	2021
market share	17.6%	31.9%
revenue	$5.3bn	$60bn
annual revenue growth	62.3%	51%

- Azure integrates with Microsoft products
- Are there jobs with Azure in Arkansas?
- Case example: Ottawa Hospital (disaster recovery)

Google Cloud Services

- Q4/2019 - pre-pandemic (changes in 2021?[5])

WHAT	2019	2021
market share	6%	9.1%
revenue	$1.8bn	$65.12bn
annual revenue growth	67%	41%

- Are there jobs with Google Cloud in Arkansas?
- Lush case example (cosmetics retailer): compensate traffic fluctuations on their e-commerce platform

Other actors

- Alibaba Cloud (China) - 5.4% (2021: 8.9%) - AI focus
- IBM - 1.8% - SPSS (stats) + Watson (health care)
- Oracle Cloud services (owns open source MySQL, VirtualBox)

How does one pick a cloud provider?

- Current infrastructure costs
- Cost of running a data center
- Cost of applications
- Cost of hiring cloud specialists
- Benefit for company and customers
- Risks (regulation, security, disaster, lock-in)

What's the right order (from most important to least important)?

Future of the cloud

- Data trends (cp. 2022 Data trends by DataCamp)
- Cryptocurrencies (Smalley, 2021).
- Post-pandemic effects

After finishing a DataCamp course:

1. download the slides and review if necessary
2. download the certificate and add it to your resume

Figure 23: Statement of Accomplishment (DataCamp, 2022)

Glossary = the learning dictionary for your brain

- Why is it important to have a glossary?

TERM	MEANING
Meta data	Data about data, e.g. control information for a database
DDL	Data Definition Language
DML	Data Manipulation Language

TERM	MEANING
DQL	Data Query Language
DCL	Data Control Language
Coupling	(design) Relates to the independence of parts of a system
gdb	GNU debugger, supports many languages
IaaS	Infrastructure as a Service (roads)
PaaS	Platform as a Service (roads + shops)
Saas	Software as a Service (roads + shops + products)
FaaS	Function as a Service
Scaling	Horizontal or vertical - increase computing performance
On-premise	Application + data reside in the company or organization
Cloud	Application + data reside in the network
Virtualization	Split the server action up across different locations
GDPR	General Data Privacy Regulation act of the European Union
Internet	Global digital infrastructure (router,network)
Web	Global collection of digital content (browser,wiki)

DB dump and output, SELECT pipeline - w6s10 - February 17, 2022

"Be the shell!": in-class assignment

See also my screencast (12 min) with script in GitHub (PDF).

1. Start SQLite with header on and column mode switched on from the command line (to find out, look at `sqlite3 --help`).

   Solution: On the command line[6], enter

   ```
   $ sqlite3 -header -column~
   ```

2. Check that you don't have a persistent database with `.database`.

   Solution:

   ```
   sqlite>  .database
   ```

3. Open your (existing) database `sqlite.db` with `.open`

   Solution:

```
sqlite>  .open sqlite.db
```

This database does not need to exist yet - if it does not, the binary file is created as an empty file in the directory where you were when you opened SQLite.

4. Check that you're now writing to `sqlite.db`

   Solution:

   ```
   sqlite>  .database
   ```

   You could also enter `.show` - the last line of its output shows the currently active database. You can change databases and have as many open as you like.

5. Check that in fact header is `ON` and the mode is `column`

   - with `.show` to show all output values
   - with `SELECT`

   Solution:

   ```
   sqlite> .show
   sqlite> SELECT * FROM customer;
   ```

   This presumes that you have created at least one table named `customer`, and that it is in your database `sqlite.db`, and that the table has at least one row - otherwise nothing will be displayed.

6. Switch the output to a file with `.output sqlite.sql`

   Solution:

   ```
   sqlite> .output sqlite.sql
   ```

   `.output` is one of the settings you saw with `.show`. It shows where the output is directed. The default is `stdout`, or the screen. You have now reset this to pipe all output into a file `sqlite.sql`.

7. Dump the content of your database with `.dump`

   Solution:

   ```
   sqlite> .dump
   ```

   This writes a version of the binary `.db` file to a text file `.sql`. It contains all the information necessary to recreate the database. This `.sql` can be used to port the database. On another computer, SQLite will recreate `sqlite.db` from `sqlite.sql` by importing it with the `.read` command.

8. Switch the output back to `stdout`

   Solution:

```
sqlite> .output stdout
sqlite> .show
```

Until we redirect the output to `stdout`, every output we generate, including error messages, will be redirected to the `.sql` file.

9. Dump the content of your database again.

   Solution:

   ```
   sqlite> .dump
   ```

   Instead of dumping to an SQL file, we are now dumping the database to the screen.

10. Leave the SQLite shell and look at `sqlite.sql`

    Solution:

    ```
    sqlite> .q
    ```

    Depending on your PC (it doesn't seem to work under MacOS), the `.shell` command can be used to peek beyond the SQLite shell, and you could use this command to look at the directory listing - under Windows: `.shell DIR`.

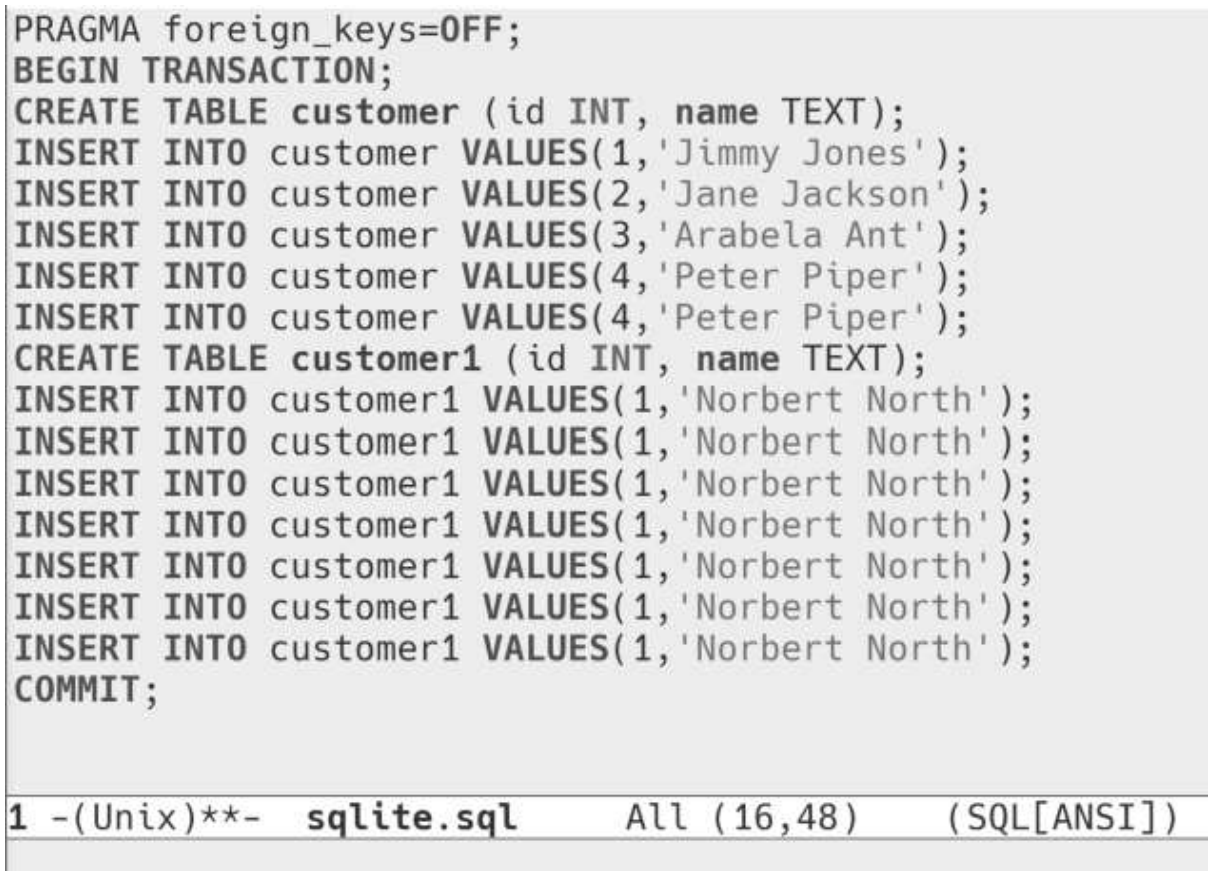


Figure 24: sqlite.sql with the sqlite.db database dump

SQLite import/export, NULL, UNIQUE - w7s11 - February 22, 2022

- Apache HTTP server still seems to be the dominant HTTP server software. XAMPP, the package featured in class, contains a fork of MySQL, MariaDB, and a GUI to use the RDBMS graphically. There are many GUIs and XAMPP is still the easiest to install, I think. XAMPP's original purpose is not for DB training though but for easy web development (you have all the tools you need on your PC). We're going to look at this in class - however, I'm not going to install XAMPP on every lab PC, so if you wish to tag along, you need to bring your laptop to class. I'm going to make an installation video.
- FAQ: It's bugged me from the start of the course that GNU Emacs on Windoze did not seem to allow to run SQLite in a shell. Turns out that's not true! The issue is merely cosmetic: the SQLite shell prompt is not shown is all.

- Here are a couple of screenshots both from Windoze - the first is the SQLite shell inside an terminal Emacs shell (start Emacs from the CMD line with `emacs -nw`), the second is from within the Ubuntu app (20.04 LTS), also the terminal Emacs.

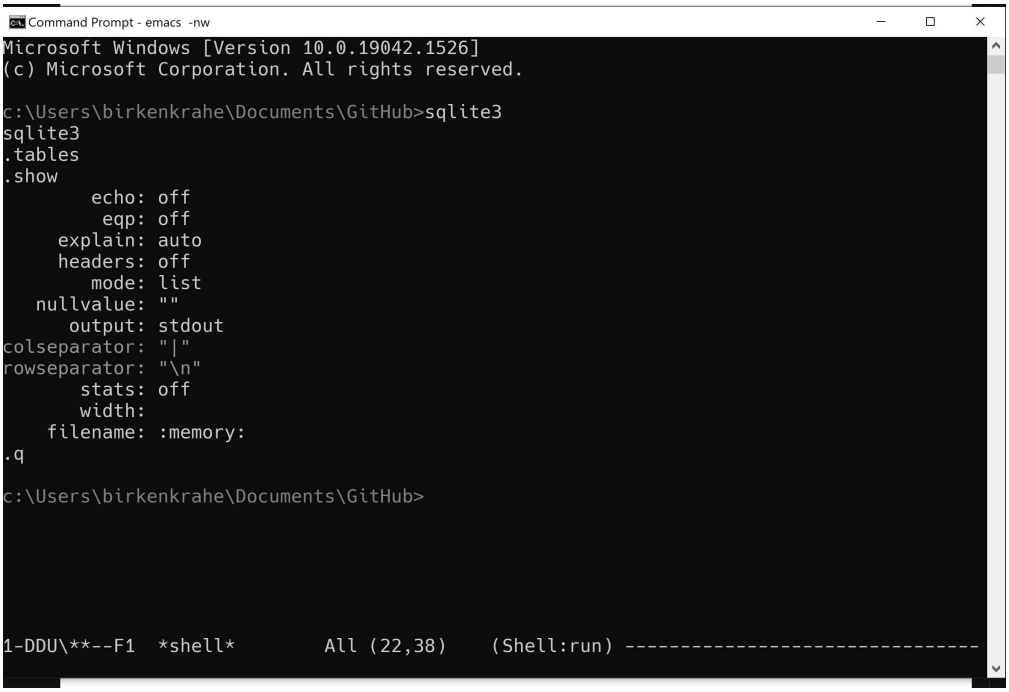


Figure 25: Terminal GNU Emacs in Windows

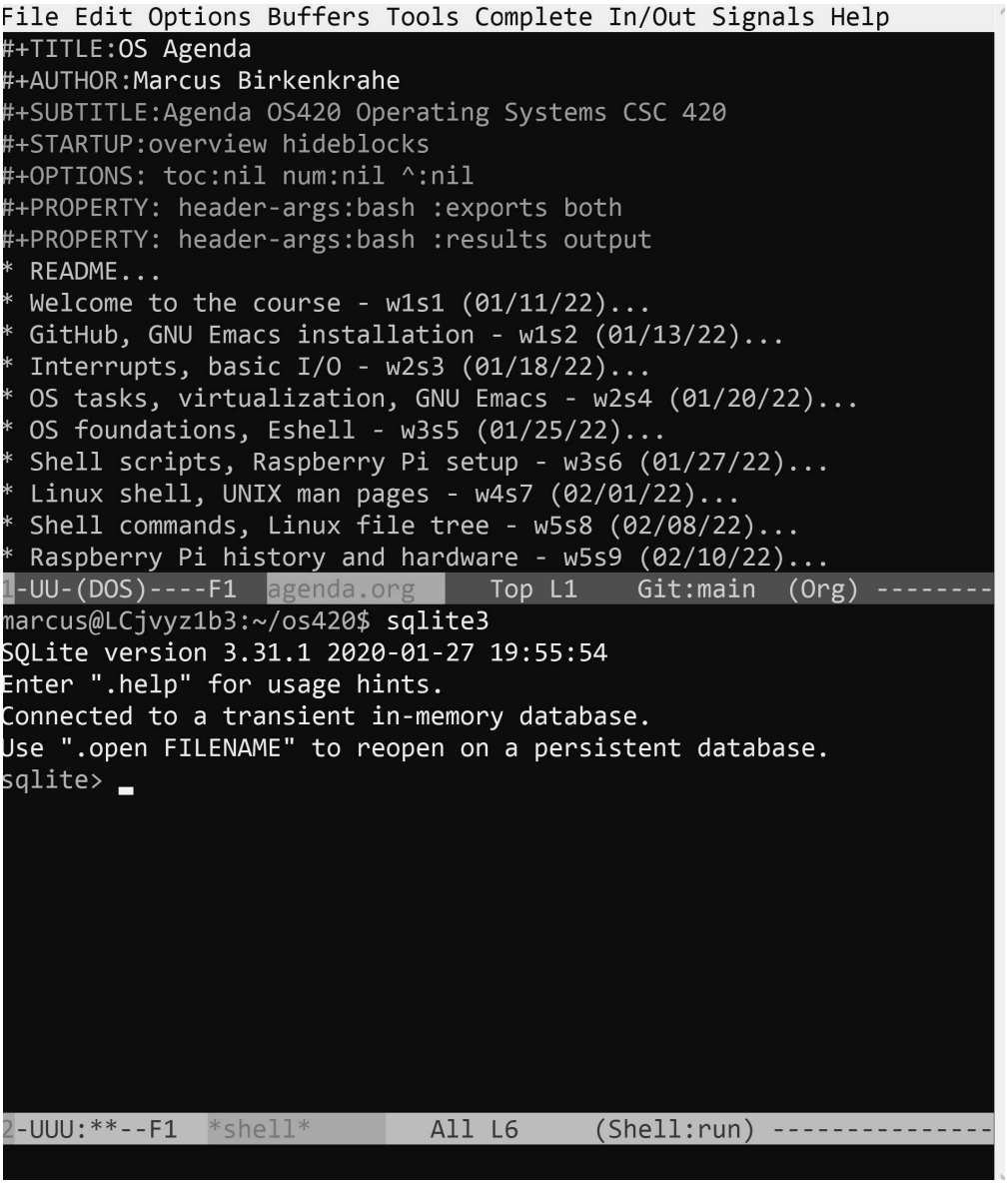


Figure 26: Terminal GNU Emacs in Windows' Ubuntu app

Midterm grades, SQLite cloud REPL - w8s13 - March 3

- If you want to improve your grade, you can talk to me about doing a small, independent research project leading to a writeup in the form of a notebook, or a short (10-15 min) presentation. The topic must be related to the topic of this course.
- The physicist I was talking about was Harry Lehmann (1924-1998), one of three fathers of the LSZ reduction formula. He was already retired when he ran the physics tutorial that I attended (if you follow the Wikipedia link to the formula, you will understand why we needed a tutor, and why it was great to have him). Lehmann was a terrible chain smoker, and the seminar couldn't stop him - it was often hard to see him through the clouds of smoke. His pupil Gerhard Mack later became my PhD supervisor. In science, this constitutes a "grandfather" relationship, which I am quite proud of even though I left particle physics long ago.

- explains it well (docker.com):

> "A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another."

The concept is similar to a Java VM runtime environment: but instead of just running Java, you can run pretty much anything in the container. As the figure shows, the Docker separates the app from the Operating System (Linux, Windows, MacOS). This is convenient, because now you don't need to bother with the OS. But it also stops you from learning anything about how apps interact with the system itself. It's super cool if all you are about is building apps, especially web apps, like replit.com. It's not so cool if you're up against legacy systems (old software or hardware), or if you actually like interacting with the OS (via the shell), or if you want to create anything new, or if your mojo is performance improvement (e.g. making algorithms or data pipelines faster), because that depends on deeper knowledge. The good news: everyone can install a container, and they safe (actually, that's another problem…more layers, more potential attack points). Here is a list of 6 issues (Brandon, 2021).

Long story short: important concept and technology, you should try it out and explore it a little, perhaps you fall in love, and in the least you get another marketable skill.

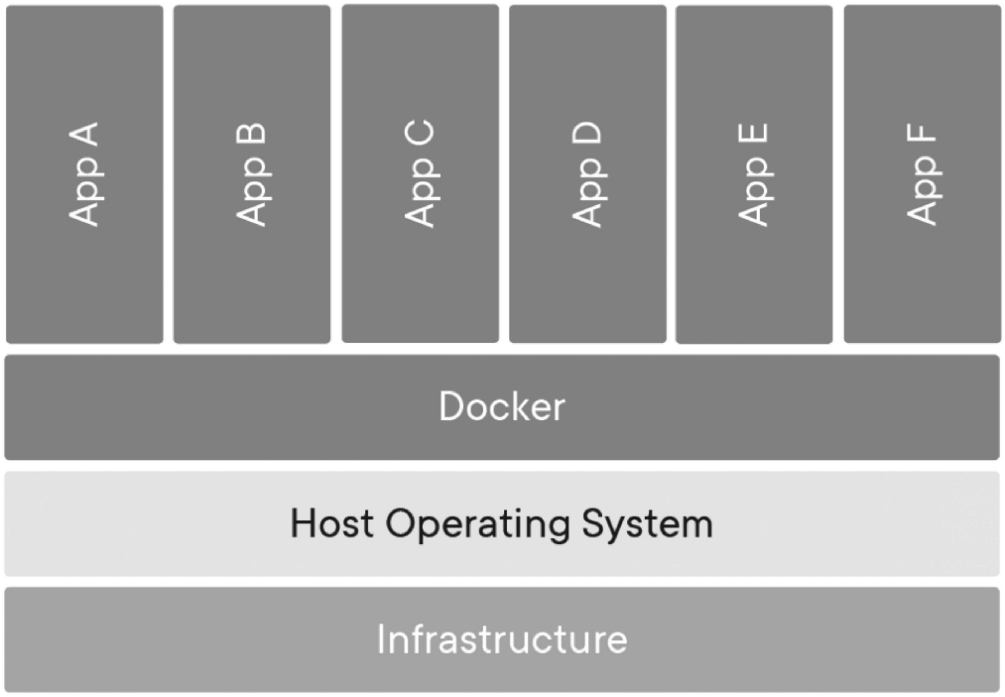


Figure 27: Containerized applications (docker.com)

- You can imagine an `INNER JOIN` operation between two tables as a zipping together of the tables using any two columns. If these two columns are primary keys (= columns that uniquely identify every row of their table), then the primary key of the new table that exists only for querying purposes, consists of not one, but two columns.

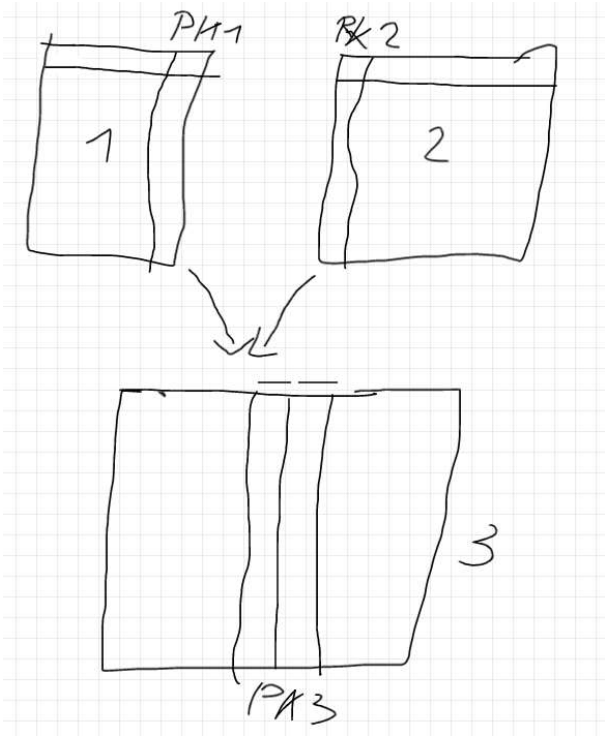


Figure 28: PK1 of 1 and PK2 of 2 become PK3 = (PK1, PK2) of 3

- The `INNER JOIN` works also if the two columns are not primary keys. E.g. in the statement shown in 1, all rows for which the condition is fulfilled that the value of `a` in the table `x` equals the value of `b` in the table `y`, are selected. For code examples, see the notebook `SELECT_xyz_nb.org`.

```
SELECT * FROM x INNER JOIN y ON x.a = y.b;
```

References

- BBC (11 Mar 2019). World wide web vs. internet - what's the difference? [video]. URL: www.bbc.co.uk.
- Birkenkrahe (Jan 11, 2022). Interactive shell vs. interactive notebook (literate programming demo). URL: youtu.be/8HJGz3IYoHI.
- bnewall1 (Apr 25, 2010). Star Trek: Captain's Log - 11/30/1994 - 2/7 [video]. URL: youtu.be/T2bSMLEQX1o.
- Brandon (Apr 10, 2021). 6 Problems with Container Technology [blog]. URL: ondat.io.
- Smalley (22 Oct 2021). Cryptocurrency is changing the Data Center Market [blog]. URL: datacenters.com.
- Harrison Metal (2019). Thank You, Barbara Minto [video]. URL: vimeo.com/305393045.
- Lucidchart (2022). What is a Database Model [website]. URL: www.lucidchart.com.
- Lucidchart (Feb 7, 2018). UML Use Case Diagram Tutorial [video]. URL: youtu.be/zid-MVo7M-E.
- Moocall (Nov 27, 2020). Moocall Calving Sensor [video]. URL: youtu.be/718uGYbUmao.
- NextCloudPi Documentation (2019). Welcome [website]. URL: docs.nextcloudpi.com.
- Nystrom (2021). Crafting Interpreters. Genever Benning. URL: https://craftinginterpreters.com/
- orgmode.org (n.d.). SQLite Source Code Blocks in Org Mode [website]. URL: orgmode.org.
- Russell/Norvig (2021). Artificial Intelligence - a Modern Approach (AIMA). Pearson. URL: aima.cs.berkeley.edu.

- Spajic (Jan 29, 2018). Understanding Git - Data Model. URL: medium.com/hackernoon.
- Valencia (Feb 28, 2017). The Lisp approach to AI (Part 1). URL: medium.com/ai-society.

Footnotes:

[1] The term "vanilla" refers to the fact that this is the uncustomized, original version of Emacs. For large, old open source projects, this is a bit of a mystery, though, since so many versions exist that it may be difficult to identify "the original".

[2] "Logic, logic…I'm sick to death of logic!" (Star Trek: Captain's Log)

[3] There is no federal data privacy law like GDPR in the United States. There are some national laws that have been put in place to regulate the use of data in certain industries, e.g. The 1974 U.S. Privacy Act (rights and restrictions regarding data held by US government agencies), or the 2018 California Consumer Privacy Act (rights and protection for CA residents).

[4] Privacy is a political term. Private information can also be an entry to phishing or other network-based activities compromising the security of your PC.

[5] Statista numbers (no source given) for 2021.

[6] Org-mode has 11 different header arguments (see here) - you can initialize SQLite with the header shown below. A separate session buffer is not supported[7] - SQLite is run outside of Emacs and the results are displayed here. For a session demonstrating SQLite commands (which begin with a `.`), Org-mode is not well suited. For SQL commands, it is well suited.

```
#+begin_src sqlite :db ./sqlite/sqlite.db :header :column :results output
  ...
#+end_src
```

[7] Such a session buffer opens e.g. when you start R with `M-x R`.

Author: Marcus Birkenkrahe
Created: 2022-03-11 Fri 22:28
Validate