DB Agenda

Agenda DB330 Database Theory and Applications

README

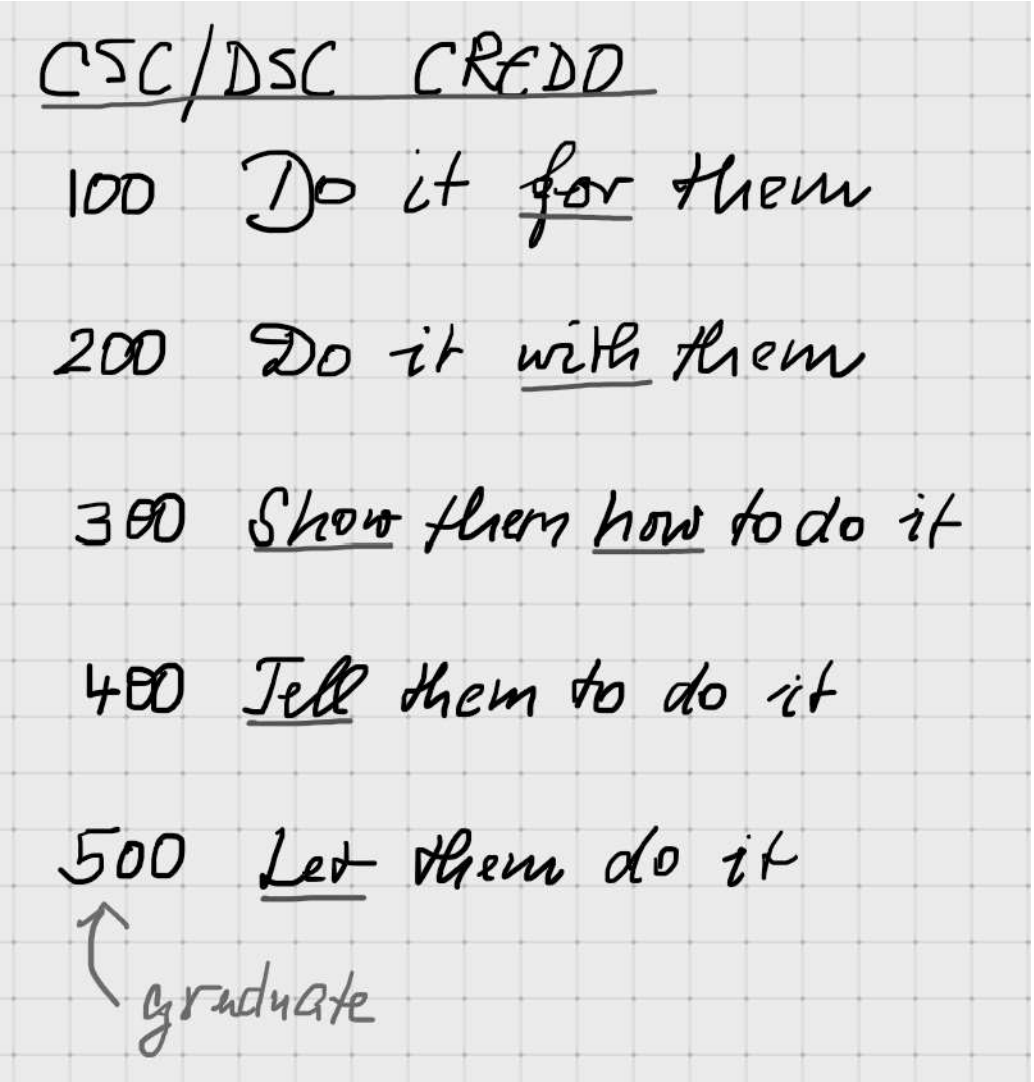
This file contains the agenda overview (what I had planned), the objectives (what we managed to do) and (much of the) content of each taught session of the course. I want to avoid splitting the content up over many files - so that you have to navigate as little as possible (like a book)!

The companion file to this file, less structured and with the captain's log, is the notes.org file.

Intro and GitHub - w1s1 (11-Jan)

- Aspiration and ambition (Lyon Data Science program)
- Introduction to the course & the lecturer
- Homework assignments: **GitHub**, DataCamp, Emacs
- What's next?

My new didactic credo



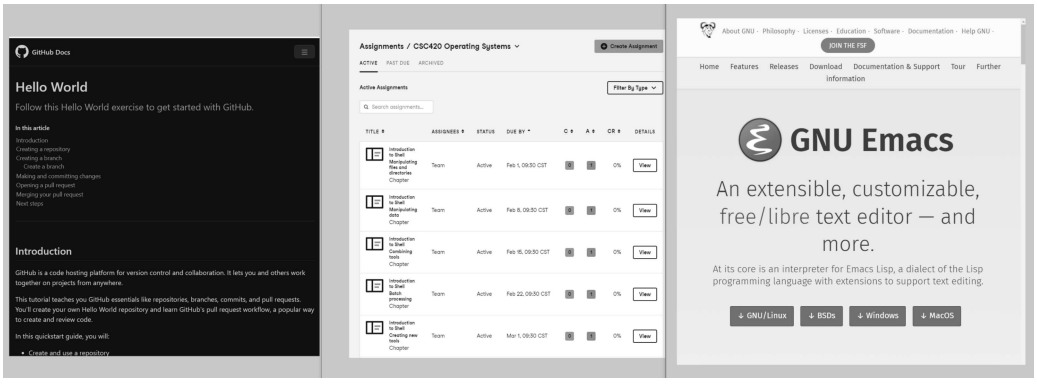
Aspirations and ambitions (DS program 2021-2023)

CLASS	CODE	TERM	Topics
Data Science Tools and Methods	DSC 101	Fall 2021	R, Basic EDA, Base R
Introduction to Advanced Data Science	DSC 205	Spring 2022	R, Advanced EDA, Tidyverse
Database Theory and Applications	CSC 330	Spring 2022	SQL, SQLite
Operating Systems	CSC 420	Spring 2022	Bash, awk, sed, regular expressions
Applied Math for Data Science	DSC 482/MTH 360	Fall 2022	Probability, Statistics + R
Data Visualization	DSC 302	Fall 2022	D3, Processing, Javascript, Bokeh
Machine Learning	DSC 305	Spring 2023	Predictive algorithms, neural nets
Digital Humanities	CSC 105	Spring 2023	Data science applications

Introduction to the course & the lecturer

- PhD theoretical particle physics / WWW development
- SQL since 2005 (Why? Particle data = unstructured)
- Professor, Business Informatics @Berlin Univ
- Visiting Assoc Prof for Data Science @Lyon (2021-23)
- Syllabus for this course (Schoology)

Homework assignments week 1 (11-Jan/13-Jan-2022)



- **GitHub Hello World Exercise (Info: FAQ) - by Thursday 13-Jan!**
- DataCamp platform registration (Link: Schoology)
- GNU Emacs installation (Info: FAQ)

GitHub

- What is it?
  - Software development platform (like GitLab, BitBucket, SourceForge, etc.)
  - Built around Git by Linus Torvalds
  - Bought by Microsoft in 2018 (like OpenAI - home of GPT3)
  - 77 mio users (developers) + 200+ mio software projects

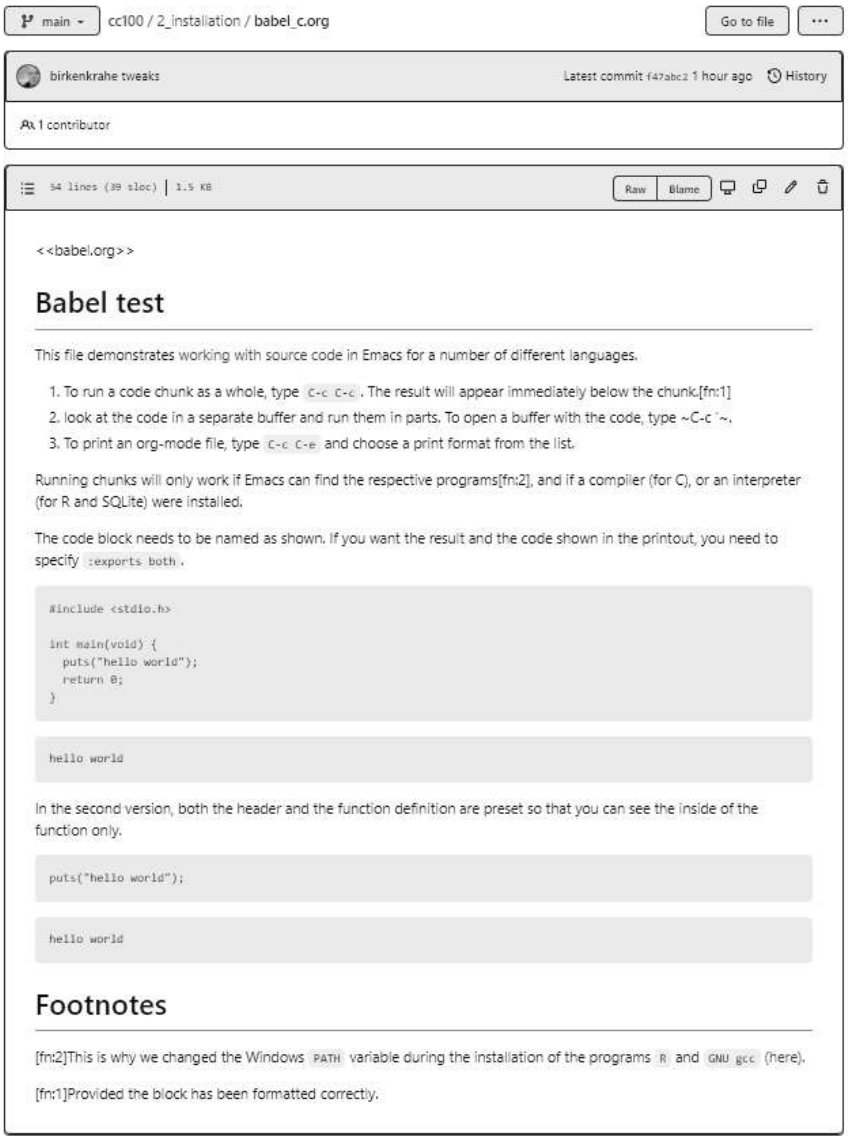
- AI support (e.g. GitHub Copilot)

Watch: "What is GitHub?" (GitHub, 2016)

Gif: "So long binder of requirements" Source: GitHub

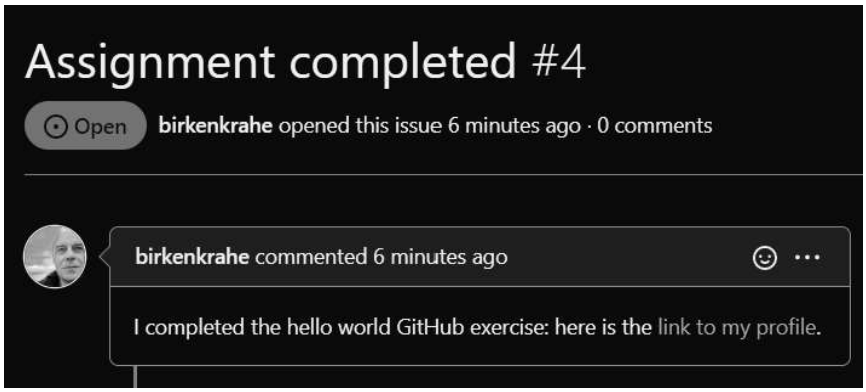
- Why are we using it?

  Image: Org-mode file in GitHub



  - It's free
  - To host course materials
  - Upload assignments (esp. Emacs Org-files)
  - Discussion
  - Wiki for collaboration
  - Complements Schoology
- What will you have to do?
  - Sign up with GitHub - use Lyon Email
  - Pick an available username **using your own first and last name**, e.g. `MarcusBirkenkrahe`, or `DonaldTrump`
  - Complete the "Hello World" exercise (FAQ)
  - Give me your GitHub username so that I can add you as a collaborator to my private `db330` repository
  - Create an issue from the `db330` repository like in the example below (except from your account instead of mine).

Image: Issue "Assignment completed"



If you do have a GitHub account already, do the exercise anyway using your existing account (it takes 10 min)! Make sure you let me know what your user name is so that I can add you to my repo.

- What else can you do?
  - You can fork the `db330` repository
  - You can watch the `db330` repository - and set `Notifications` to `Participating and @mentions` so that you see my comments (see image below).

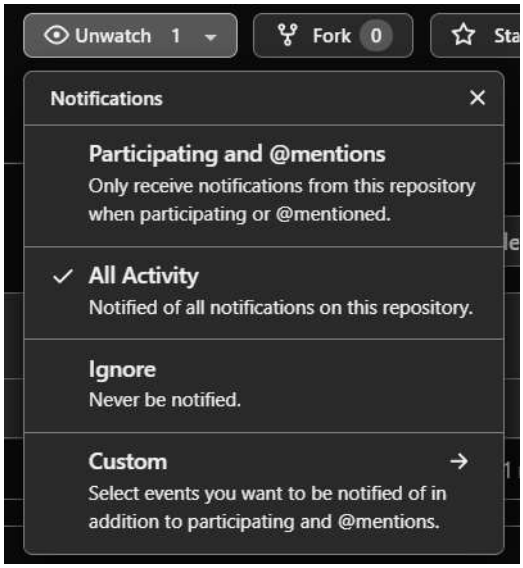
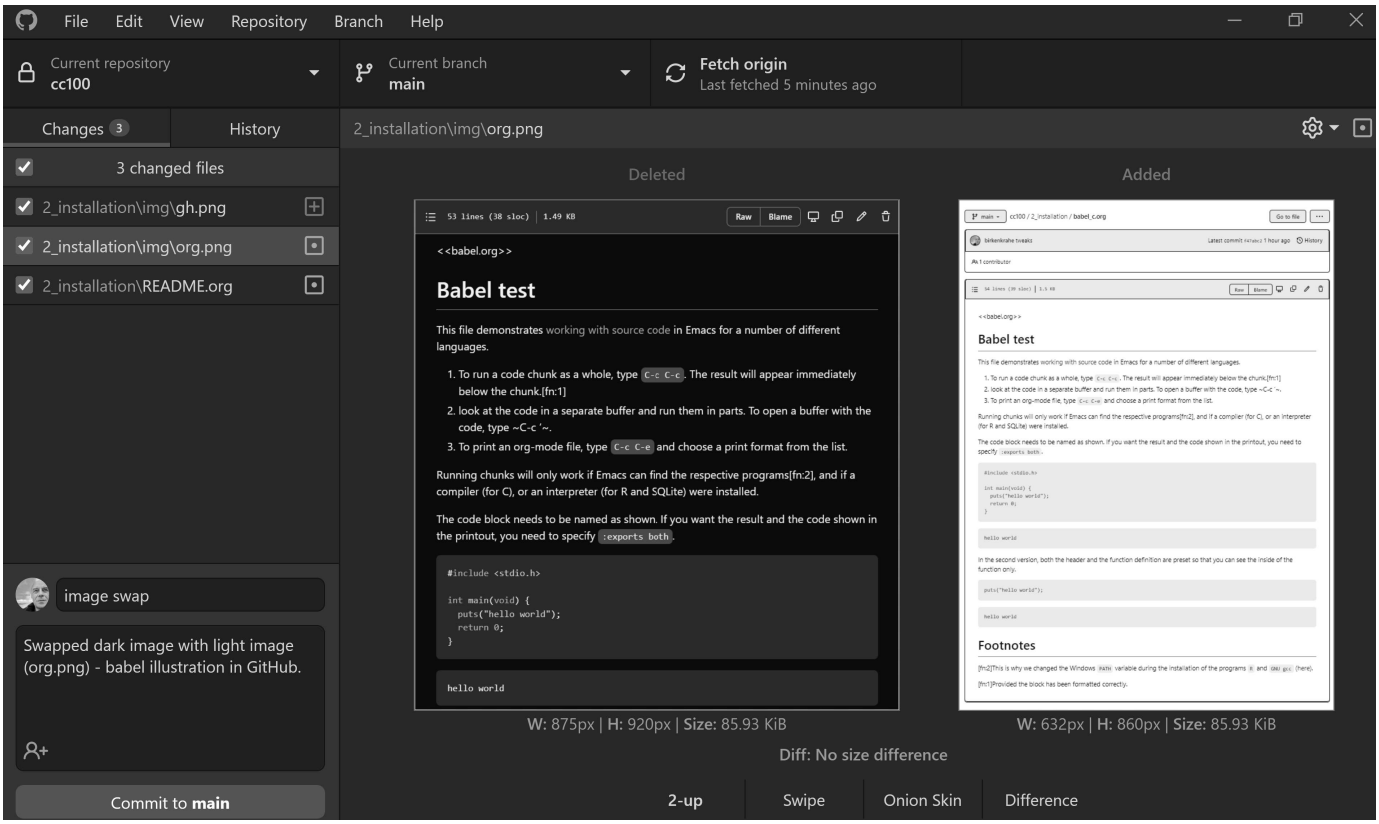


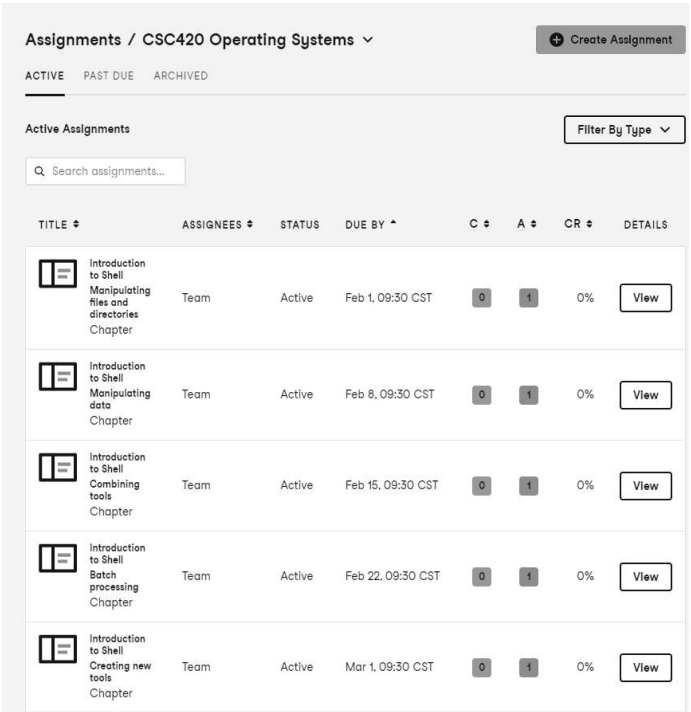
Image: Notifications settings when watching a repository

  - You can submit issues from the repository (e.g. if you notice mistakes or if you want extra information, or to share a link)
  - You can participate in discussions (sometimes I will make you)
  - You can add to the wiki (e.g. comments and links to interesting resources)
  - You can install the mobile app on your smartphone
  - You can use it as a platform for projects or coding
  - You can download the desktop client to manage repos on your PC (see image below).

Image: GitHub desktop client commit



DataCamp



- Why are we using it?
- How are we using it?

- What will you have to do?

GNU Emacs (1976…1985)

- Why are we using it?
  - To mix documentation + code + output = literate programming (1984)
  - It's the same thing as an interactive computing notebook (Jupyter) … except open to ALL languages and outputs
- How are we using it?
- What will you have to do?

What's next?

- See schedule (GitHub)
- Watch online lecture on "Systems"
- Everything else = online summary
- See you (hopefully) Thursday in class! (Lyon 104)

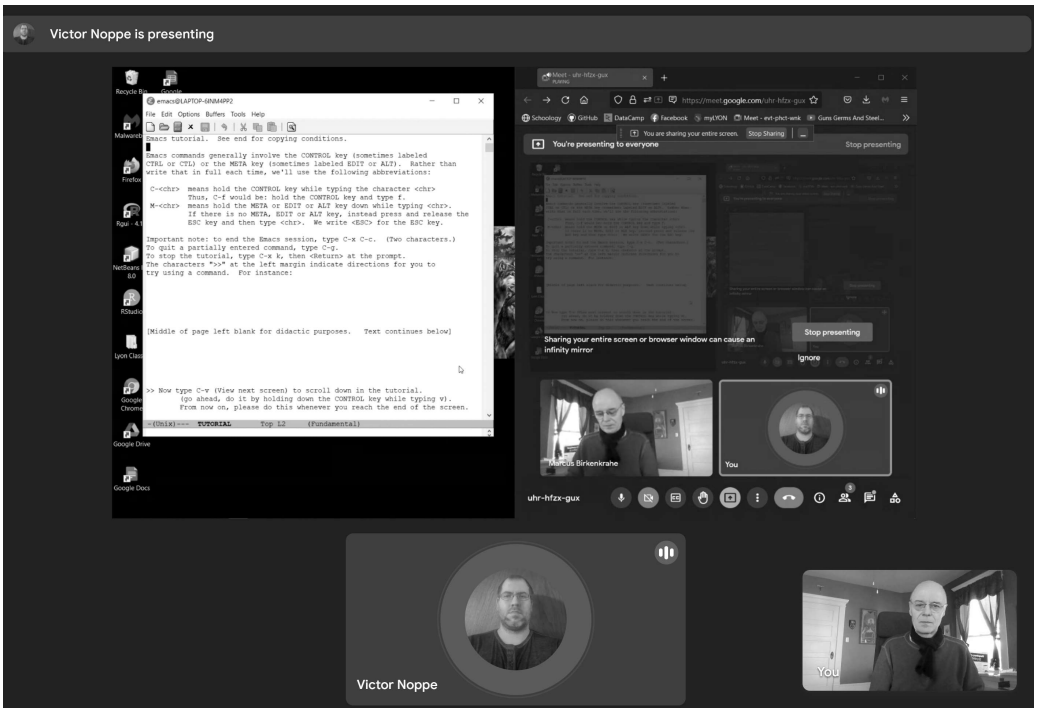


Figure 10: Victor N. installs Emacs @10:50 AM

DataCamp, History of DB, MooCall - w1s2 (13-Jan)

Overview

HOW	WHAT
Review	GitHub Hello World exercise (FAQ)
Video	History of Databases (Codd, Oracle, IBM)
Lecture	DB applications and basic definitions
Application	MooCall Calving Sensor App (IoT network)
Practice	Install GNU Emacs (ESS or vanilla) (FAQ)
Demo	Emacs guided tour
Self	Work through the Emacs onboard tutorial

Objectives

- [X] Review the basics of Git and GitHub
- [X] Know basic definitions of database management systems
- [X] See a current example of an IoT-based DB application
- [X] Install the GNU Emacs editor on your OS
- [ ] Understand how GNU Emacs works
- [ ] Make GNU Emacs work for you

DB elements, GNU Emacs - w2s3 (18-Jan)

Overview

HOW	WHAT
Review	Quiz 1: Intro to course / databases
Lecture	Elements of Database Systems
Demo	Emacs guided tour
Self	Work through the Emacs onboard tutorial

Objectives

- [X] Review last week's content with a quiz
- [X] Review file vs. db approach to data management
- [X] Learn about the elements of a database system
- [X] Understand DB system design structure/users
- [X] Understand how GNU Emacs works (guided tour)
- [ ] Make GNU Emacs work for you

What's next?

- Take a look at the Emacs tutorial (CTRL-h t)
- GNU Emacs practice exercises (in class)
- DataCamp assignments beginning next week (online)

SQLite installation - w2s4 (20-Jan)

Overview

HOW	WHAT
Demo	Installing `sqlite`[1] (done[2])
Practice	Emacs guided tour (tour)(web-tutorial)
Self	Work through the Emacs onboard tutorial

Objectives

- [X] Understand how GNU Emacs works (guided tour)
- [X] Make GNU Emacs work for you (tutorial)

What's next?

- Architecture and classification of databases
- DataCamp assignments due next week (online)
- Cloud computing - relevance for databases

Cloud Computing for Everyone Cloud Computing for Everyone Introduction to Cloud Computing I forgot to mention this in the last class. This is the first DataCamp assignment. It is informational and very simple - nothing but drag and drop practice and a few videos. This should not take you longer than 15-20 minutes.

We'll pick up on past assignments in class - short review including questions for the audience (you!) Completing this assignment on time gets you 10 points (100%). Late completion (after the due date): 5 points (50%).

Cloud computing intro - w3s5 (25-Jan)

Overview

HOW	WHAT

HOW	WHAT
Review	Quiz 2 - database foundations / Emacs
	DataCamp assignment 1: cloud computing intro
Quiz (opt)	Inside Google's Data Center (Thu)
Demo	Emacs Org-/ code blocks
Practice	Create literate Org-mode file
	Run `sqlite` program in Emacs
Assignment	Do this on your PC (extra credit by Thu 1PM)

Objectives

- [X] Review db foundation and GNU Emacs (quiz 2)
- [X] Review introduction to cloud computing (DataCamp 1)
- [X] Understand GNU Emacs Org-mode
- [X] Know how to create a literate Org-mode file
- [X] Know how to run a literate Org-mode file

DataCamp assignment: Cloud Computing Introduction

- What's the main message of this lesson?
- What does cloud computing have to do with databases?
- What did you think about the assignment?
- What did I think about the assignment? (see notes)

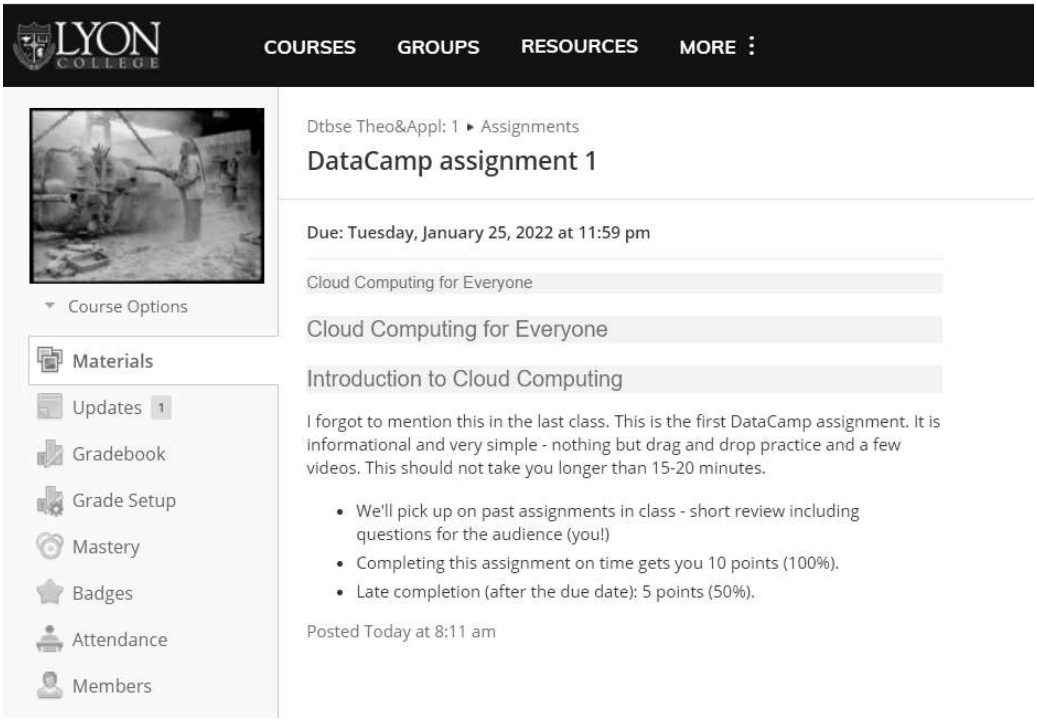


Figure 11: Schoology assignment, Jan 21, 2022

What's next?

- Architecture and classification of databases (book)

- Next DataCamp assignment due Feb 1 ("Cloud deployment")
- Org-file assignment (in Schoology) coming your way

Cloud deployment - w4s6 (1-Feb)

Overview

HOW	WHAT
Review	Quiz 3 - data centers / cloud computing / metadata
	DataCamp assignment 2: cloud deployment
	Emacs Org-mode assignment (see tutorial videos)
Lecture	Database foundations - 10 tenets
Practice	SQLite basics - creating/importing a database

Objectives

- [X] Review quiz 3 - how should you study, learn and rehearse?
- [X] Review deployment of cloud computing services (DataCamp 2)
- [X] Review database foundations (10 tenets)
- [X] Opened, closed SQLite (DBMS) and wrote on a db (see notes)

DataCamp assignment: Cloud Deployment

- What's the main message of this lesson?
- What infrastructure is required? Can you do this yourself?
- What're the greatest challenges of deployment?
- What did I think about the assignment? (notes)

Database management foundations - 10 tenets

1. A database (DB) is a collection of related data items within a specific business process or problem setting.
2. A database management system (DBMS) is the software package used to define, create, use and maintain a database.
3. We distinguish the file-based vs. DB approach to data management (data from different application stored in different files vs. managed by one application and one shared, central database)
4. Metadata are data (structure) definitions, like ownership or number of tables, and are stored in the DB catalog or dictionary.
5. DBMS provide DB languages (like SQL) that facilitate data definition (DDL), data manipulation (DML), data querying (DQL), and data control (DCL).
6. The database model (or schema) describes the DB data structure. It does not change easily, and is stored in the catalog. Examples are: entities stored (e.g. as tables), and entity aspects (e.g. as columns). To model, we use ERDs.
7. The database state, or set of instances, represents the data in the DB at a given moment in time, viewed using the DB language (DQL). Examples are DB records (rows) from subsequent observations.
8. DB follow a 3-layer design: an internal technical layer (e.g. server, file, network organisation), a conceptual/logical layer (e.g. the schema), and an external layer (views of the data provided e.g. by SQL queries).
9. There are different types of DB users, with different skill sets: information (cloud) architects; DB designers (ERD); DB Administrators (SQL, Linux); Application Developers (e.g. Web App); Business Users (SQL).

Practice: SQLite Basics

We'll learn more about SQLite in future sessions. This is just to get our feet wet, including some important file system aspects.

```
$ sqlite3
sqlite> .database
sqlite> .q
$ touch test.db  # this only works if you have 'touch' installed, e.g.
                 # via the cygwin utility bundle
$ fsutil file createNew test.db 0 # creates an empty file in ./
$ sqlite3 test.db
```

```
sqlite> .database
sqlite> .q
```

As you can see, it's never easy to do anything in Windows. We're better off writing SQLite code in Emacs where the `.db` file is automatically created (see assignment). If this doesn't work for you, contact me and we'll sort you out!

What's next?

- ~~Architecture and classification of databases (text book ch 2)~~
- Next DataCamp assignment due Feb 8 ("Cloud providers/cases")
- New DataCamp assignment due Feb 15 ("Introduction to SQL: SELECT")
- SQLite DDL practice

<<<<<<< HEAD

Cloud providers, SQLite introduction - w5s7 (8-Feb)

Overview

HOW	WHAT
Review	Cloud Providers and Case Studies
Lecture	Introduction to SQLite
Practice	Exploring `sqlite3`
Test info	Test 1 on Thu 10 Feb 1.30-2.15 pm

Objectives

- [ ] Understand setup for test 1 (online in class)
- [ ] Review DataCamp assignment
- [ ] Understand what SQLite is and why it's important

Test 1 info

- Online in Schoology, Thu 10 January 1.30-2.15 pm
- Quiz 1-3 are not visible during the test
- The 10 hardest questions of quiz 1-3 (< 50%)
- 10 brand new questions
- Maximum time = 45 min

DataCamp assignment: Cloud Providers and Case Studies

- What's the main message of this lesson?
- How does one pick a cloud provider?
- What did I think about the assignment?

Introduction to SQLite

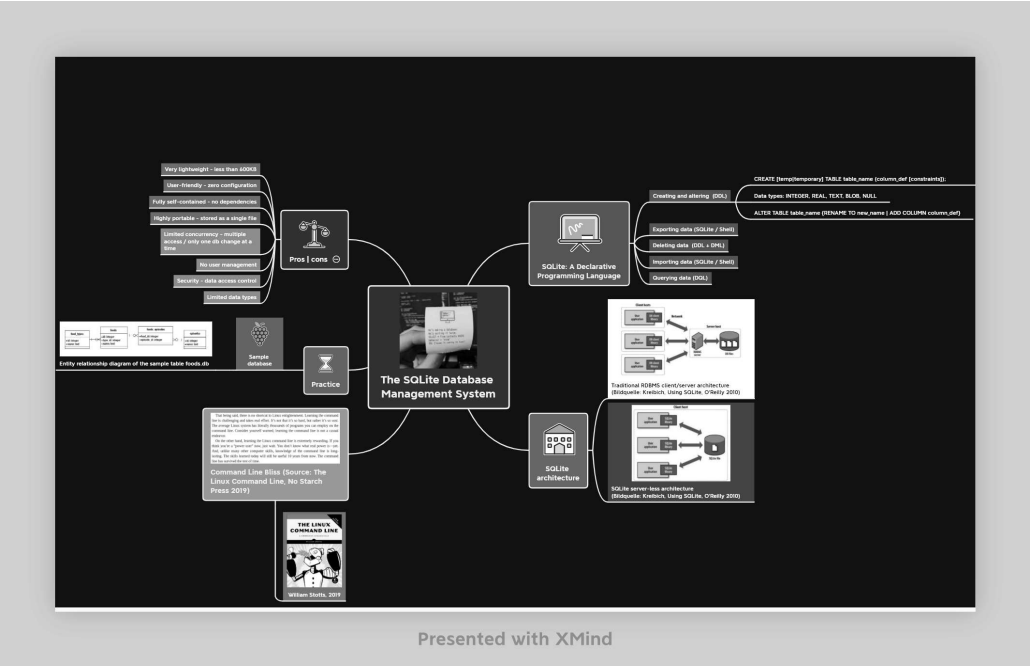


Figure 12: Introduction to SQLite (mindmap)

IN PROGRESS Practice: SQLite Basics (cont'd)

- We last looked at entering SQLite (notes)
- Today, we look at a few more commands:

```
$ sqlite3 sqlite.db  # starts SQLite on the Command shell
$ sqlite3 -help      # options list
sqlite> .databases   # prints current persistent database
sqlite> .show        # display and I/O options
sqlite> .tables      # check for tables
```

What's next?

- Test 1 - Thursday 1.30-2.15
- New DataCamp assignment due Feb 15 ("Intro to RDBM with SQL")
- SQLite DDL practice

SQLite lab session, test 1 - w5s8 (10-Feb)

Overview

HOW	WHAT
Interactive Lecture	Introduction to SQLite 2
Test 1	Thu 10 Feb 1.30-2.15 pm (online)

Objectives

- [X] More SQLite command line practice
- [X] SQLite lab session

SQLite Lab session

You should type the commands shown here into your own SQLite shell to get some practice -

- To open SQLite under Windows, open a CMD shell and type `sqlite3`. If this does not work, you either have not installed the program, or you don't have the location of the executable `sqlite3.exe` in the `$PATH` of your PC.
- To find the `$PATH` to the executable, open an `*eshell*` in Emacs (`M-x eshell`) and type `which sqlite3` (that's a Linux shell command, and Emacs simulates Linux shell commands using Emacs Lisp).
- Inside SQLite, on the SQLite shell (indicated by the prompt `sqlite>`), you need to operate with a persistent database file that must have the ending `*.db`.

- The output to the code in 1 shows 1) we have a persistent database that we're currently writing to, 2) where the file is, 3) we have no tables (yet).

```
.database
.tables
```

```
main: c:\Users\birkenkrahe\Documents\GitHub\db330\sqlite.db r/w
```

- To create a table, we need SQL commands (not just SQLite shell commands). In 1, the SQL keywords are all capitalized though SQLite doesn't actually care about that.

```
CREATE TABLE customer (id INT, name TEXT);
```

  - `customer` is the name of our table
  - `id` and `name` are the two columns of our table (stuff we wish to store)
  - `INT` and `TEXT` are two SQLite data types

- `CREATE TABLE` is a DDL (Data Definition Language) command. We now have a table but no content.

```
.tables
```

- `INSERT INTO` is a DML (Data Manipulation Language) command. We insert two lines. The result is silent.

```
INSERT INTO customer VALUES (1,"Jimmy Jones");
INSERT INTO customer VALUES (2,"Jane Jackson");
```

- To look at what we've entered, we use the `SELECT` command, a DQL (Data Query Language) command.

```
SELECT * FROM customer;
```

  - This command is best understood as a data pipeline (more later)
  - Its general form is `SELECT [cols] FROM [table] [filter]`
  - The wildcard symbol `*` means "all columns" of the table

1	Jimmy Jones
2	Jane Jackson

What's next?

- New DataCamp assignment due Feb 15 ("Intro to RDBM with SQL")
- SQLite DDL/DML/DQL practice continued

Review test 1, introduction to SELECT - w6s9 (15-Feb)

News

- Matthew Stewart, Stone Ward (Fri 18 Feb 3-3.50 PM) via Google Meet

Objectives

- [X] Understand test results
- [X] Know what to do different next time
- [X] Discuss selected questions and answers
- [X] Understand how "in class assignments" work
- [X] Review DataCamp lesson "Selecting columns"
- [ ] Get an introduction to SELECT

Test review

Test 1 results

- I think the results are actually very decent - > 75% is a good result - better next time!

Statistics

# of Grades	28	**Average**	15.33 (76.67%)
Max Points	20	**Standard Deviation**	2.15 (10.77%)
Highest Grade	18.67 (93.33%)	**Median**	15.69 (78.44%)
Lowest Grade	9.54 (47.71%)	**Mode**	N/A (N/A)

Figure 13: Test 1 results (Schoology)

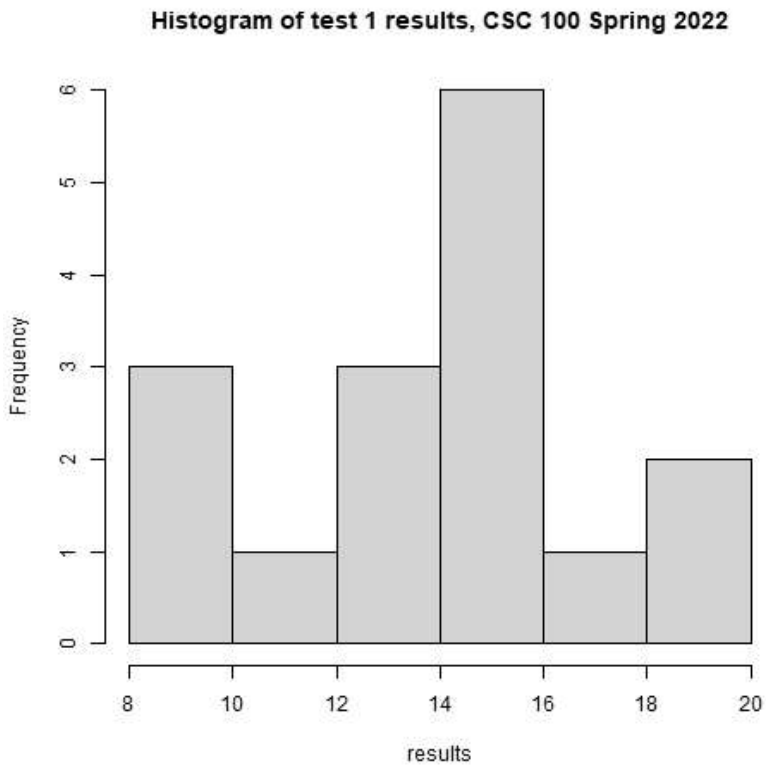
- What surprised me most was that many of you did not use the available time. Alas, the stats don't show this number. A quick glance does not seem to reveal any clear pattern.
- I am an obsessive fact-checker. When checking the stats with R, I find slightly different results:

```
results <- c(17.88,15.8,16.67,12.32,9.54,16.56,12.31,12.78,14.33,
             18.67,18,17.56,12.64,15.56,15.86,14.64,13.96,14.75,16.43,
             15.58,16.44,13.98,18.56,14.91,13.11,17,16.33,17.21)
length(results)
sd(results)
summary(results)
```

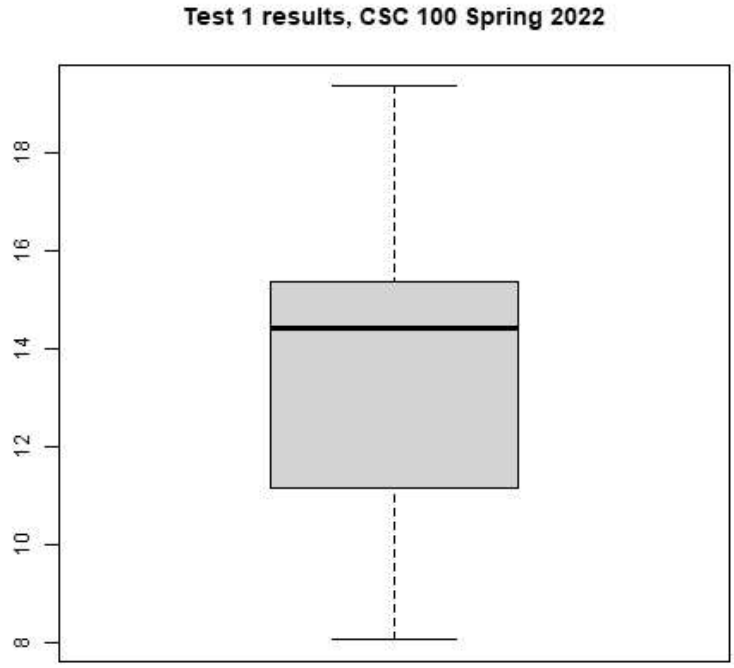
```
[1] 28
[1] 2.193788
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   9.54   13.97   15.69   15.34   16.75   18.67
```

- Let's make some plots: histogram, boxplot and density plot. I'd like the histogram and the density plot (a smoothened histogram) to peak more to the right, and for the boxplot to be smaller and higher up.

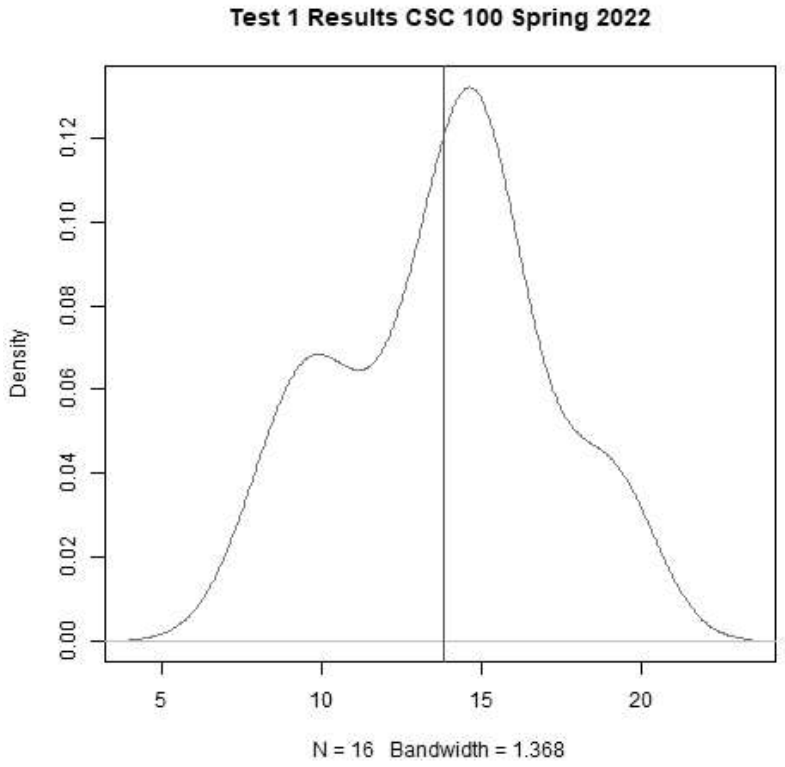
```
hist(results, main="Histogram of test 1 results, CSC 330 Spring 2022")
```




```
boxplot(results, main="Test 1 results, CSC 330 Spring 2022")
```



```
ave <- mean(results)
d <- density(results)
plot(d, col="steelblue",main="Test 1 Results CSC 330 Spring 2022")
abline(v=ave,col="red")
```



Analysis - feedback and action points

- Test 1 can now be played an unlimited number of times. I have added feedback to all new questions.
- What surprised me most was that many of you did not use the available time. However, I have not (yet) been able to correlate test time and test success.
- See also: "I can teach it to you but I cannot learn it for you"
- Questions:
  - How did you study for this test?
  - If you didn't perform well, what will you change?
  - What can I do to help you help yourself?
- Changes to be applied in future quizzes/tests:
  - Fewer multiple choices (max. 4)
  - Announce if a question has > 1 answer (and/or how many)

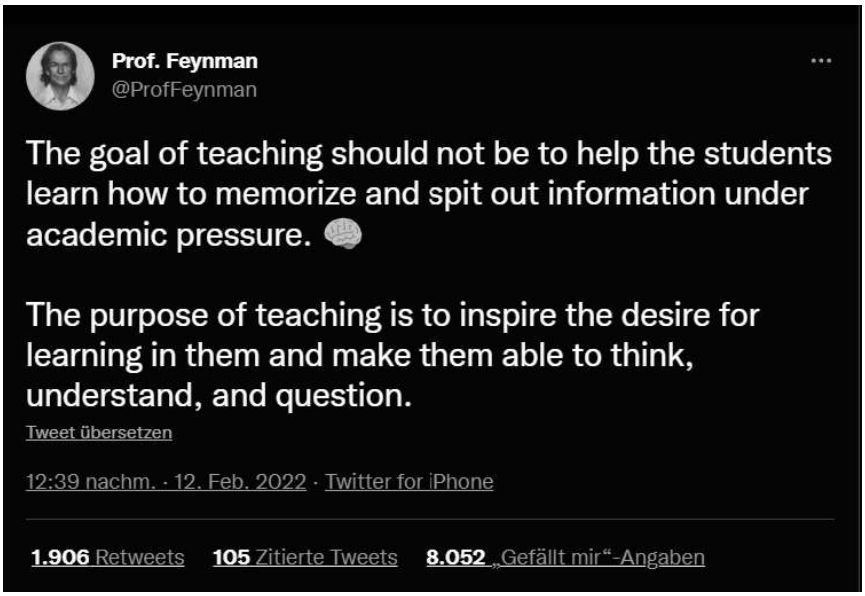


Figure 17: Feynman (via Twitter)

Individual questions

Some questions do not accept a feedback option. I have put a GitHub link into the question field.

- See the FAQ for a question on the problem of "decreasing control" for different cloud computing architectures. Some of you will get points for giving the correct order, irrespective of the direction - because indeed there was a conflict with the DataCamp image showing that "control" decreases from IaaS to SaaS, and not the other way around - because this was with regard to loss of control from complexity, for both provider and customer, while I was asking about the customer specifically. Complexity/abstraction is an important parameter in system design but less relevant for the customer, more relevant for the provider.

- Customer control for different cloud architectures (Ø=0.46)

  See FAQ - student question and answer.

- How does MooCall sensor operate? (Ø=0.96)

  Bring the following processes in the right order.

  1. Edge computing: data are generated and pre-processed locally
  2. Cloud computing: data are processed globally to generate a signal
  3. Local computing: signal is transmitted to the user

  Feedback: The sensor is attached to the cow's tail. It records data as signals in the form of temperature, motion, etc. In relation to the cloud, the sensor (a microcontroller with minimal operating and processing capabilities) operators "on the edge" (of the cloud). It has likely an SQLite database on board, or perhaps something even simpler. Gathering the data and writing them to a file if only for transport to the cloud is the "preprocessing". In the next step, the signal for the end user is prepared once the data (from different cows, and over a longer period of time) are now "in the cloud". In practice this means that they are now in a database on a server, very likely in tabular form. The processing in the cloud generates a signal for the end user when the evidence is conclusive, and when the software used to process the data has reached a positive prediction: "Cow is ready to calve". Only then the end user, the farmer, will be notified. In another scenario, the farmer is continuously informed about the state of the cow. But this does not change anything in the data processing pipeline.

- Database architecture layer and example applications (Ø=88)

  This question goes back to a slide from the DB foundation presentation (Lemahieu et al, 2021).

- The **external layer** is what the end-user sees - e.g. a manager looking at a dashboard, or using SQL to formulate a database query. The image shows entities and their attributes.
- The **conceptual/logical layer** is the database schema that translates the user requirements - stored entities and their attributes - into a database schema. The "logical" part relates to the fact that this schema needs to fulfil certain conditions to be usable by the RDBMS - e.g. records need to be uniquely identifiable.
- The **internal layer** refers to the physical implementation, e.g. the server infrastructure used to split up, or duplicate, databases over an international network.

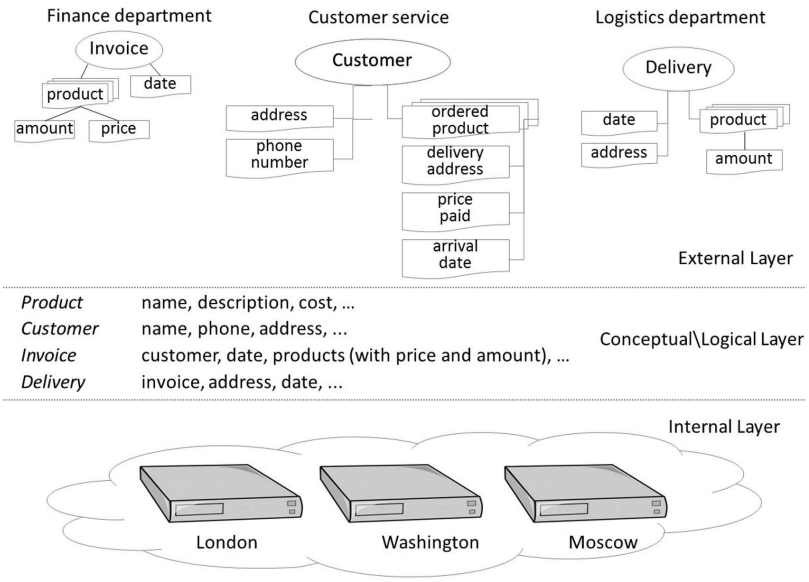


Figure 18: Three-layer DBMS architecture (Lemahieu et al, 2021)

- Pros and Cons of SQLite (Ø=88)

PROS	CONS
Very lightweight	No concurrency
Zero configuration	No user management
No library dependendies	Security - no data access control
Highly portable	Limited data types

  - See Introduction to SQLite (mindmap). This mindmap does not give an explanation (or even a source) of these properties, just a listing. For details, see e.g. keycdn (2018).
- Use case diagrams and UML (Ø=0.71)
  - UML is Unified Modeling Language - an important modeling framework for information systems design - from the small (database system) to the large (e.g. all systems of an airport). Use case diagrams are one of the 14 (standard) diagram categories of this language - and probably the easiest at that.
  - Here is a short introductory video on Use Case diagrams only.
- In research communication, "limitations" means "bias" (Ø=0.54)

  In the first video of the last DataCamp chapter on cloud computing for everyone, the presenter initially spends a lot of time trying to explain potential bias - e.g. because of the (commercial) sources of information. This bias effectively limits the validity of the presentation - high bias means that we cannot simply believe what we're told. Scientific publications must have a section on "limitations" of the research that is mainly about bias of method, and bias of the researcher.

- What are "Meta Data"? (Ø=0.89)

  Meta (from Greek, "with", or "alongside") Data come with the data, and they are always data about data - in the case of databases, e.g. about the ownership of the database. Definitions, and queries belong to DDL, and DQL, respectively. Data design issues include both of these.

  An example outside of databases are the control codes for Emacs Org-mode files beginning with the characters `#+`. They contain layout information (e.g. `#+OPTIONS: toc:nil`) or information about title or authorship.

- Database virtualization (Ø=0.61)

  Virtualization is an important principle of system abstraction: one abstracts from (= eliminates) the notion of physical location. The database appears to be in one place (at the external level) but at the internal level, it is distributed. The user never knows this.

  The same concept applies to operating systems whose processes are virtualized in the sense that jobs are executed by a concerted action of CPU, volatile memory (RAM) and non-volatile memory (e.g. Hard disk), while the user knows nothing about it.

- Entering sqlite3 at the prompt opens SQLite to a transient database (Ø=0.54)
- Which database language properties does SQLite have? (Ø=0.64)

  DDL, DML, DQL

How do class assignments work?

- In-class assignments are **10%** of your total grade
- They are labeled **class assignments** in the Schoology gradebook
- You get the points if you attend and participate **actively**
- If you check your phone instead, you're **not** active
- If you could not attend (with a good excuse), submit **late**
- Submit an Org-mode file, not a screenshot

Review of DataCamp (interactive)

- Review dashboard: SQL shell / table view

- `AS` is an alias operator: it is used in 1 as an alias for the column name.

```
.header ON
.mode column
SELECT name AS result FROM customer;
```

```
result
------------
Jimmy Jones
Jane Jackson
```

- The `AS` in DataCamp created a column. You can use `SELECT` to "pipe" anything into a table format.

```
.header ON
.mode column
SELECT 'Hello world' AS hello;
```

- Demonstration of `DISTINCT` using our table `customer`.

```
.header ON
.mode column
INSERT INTO customer VALUES (3,"Jimmy Jones");
SELECT * FROM customer;
SELECT DISTINCT name AS dist_name FROM customer;
DELETE FROM customer WHERE id=3;
```

- Demonstration of `COUNT` using our table `customer`.

```
SELECT COUNT(*) FROM customer;
INSERT INTO customer VALUES (3,"Arabela Ant");
```

```
SELECT COUNT(*) FROM customer;
```

GLOSSARY

TERM	EXPLANATION
SQL	Structured Query Language
table	Rectangular data structure, set of rows and columns[3]
record	Row for set of observations on one entity
field	Column for attribute of all rows in that table
`AS`	SQL alias operator
`DISTINCT`	Selector of unique values from one column
`COUNT`	SQL aggregator function

Next

- Overview of the `SELECT` pipeline
- More SQLite gymnastics: `stdout`, `.dump`
- DataCamp: filtering results

DB dump and output, SELECT pipeline - w6s10 (17-Feb)

Objectives

- [X] Review: Feb 15 in-class exercises
- [X] Practice: SQLite stdout and dump
- [X] Lecture: SELECT pipeline

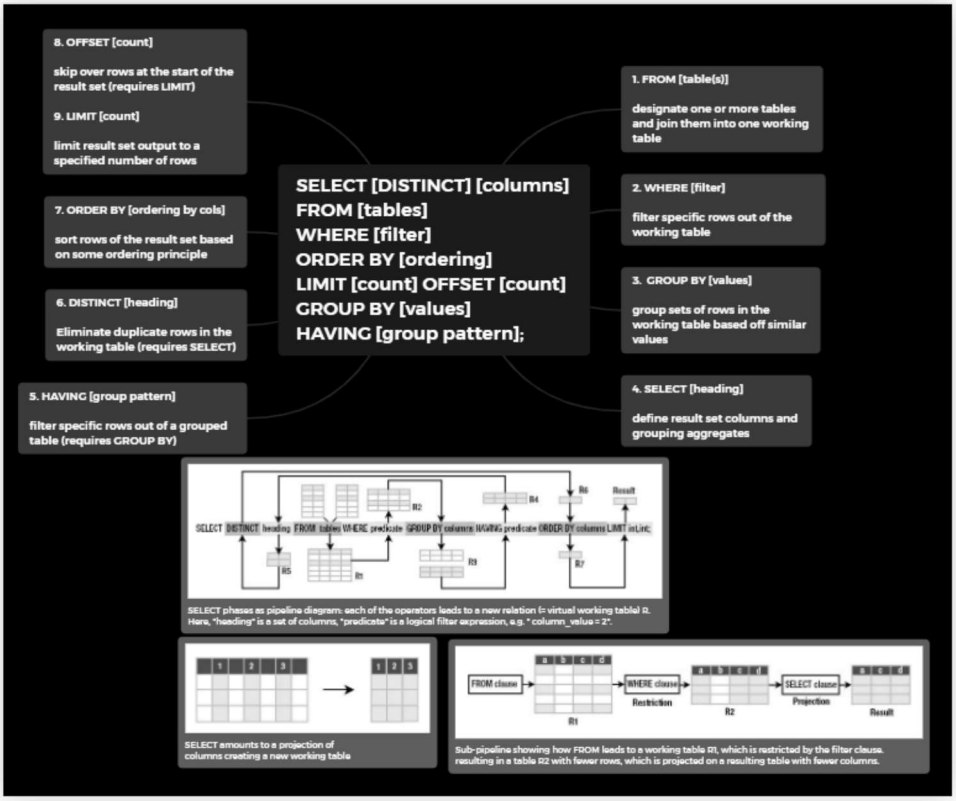
"Be the shell!": in-class assignment

You can perform all of these commands inside an Org-mode file to keep them[4], or on the command line, if you don't care about losing it - see class notes (GitHub) and my screencast on YouTube with a script (PDF).

1. [X] Start SQLite with header on and column mode switched on from the command line (to find out, look at `sqlite3 --help`)
2. [X] Check that you don't have a persistent database with `.database`
3. [X] Open your (existing) database `sqlite.db` with `.open`
4. [X] Check that you're now writing to `sqlite.db`
5. [X] Check that in fact header is `ON` and the mode is `column`
   - with `.show` to show all output values
   - with `SELECT`
6. [X] Switch the output to a file with `.output feb17.sql`
7. [X] Dump the content of your database with `.dump`
8. [X] Switch the output back to `stdout`
9. [X] Dump the content of your database again
10. [X] Leave the SQLite shell and look at `feb17.sql`

Overview of `SELECT`

- Overview of the SELECT pipeline (mindmap)
  - Pipeline of different virtual views
  - Projection of columns in tables
  - Grouping of rows by column
  - Pattern identification of rows
  - Ordering and limiting for display



Presented with XMind

Figure 19: High level overview of SELECT

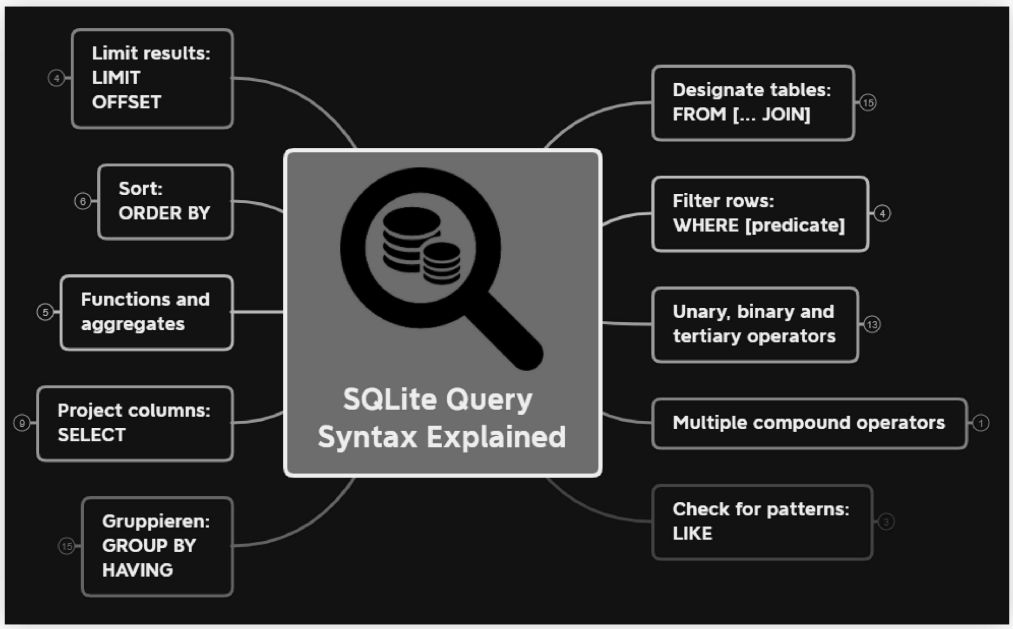


Figure 20: SELECT in detail

Next

- More SQLite (import, export, delete)
- More SELECT (joins, filter, arithmetic)

SQLite import/export, NULL, UNIQUE - w7s11 (22-Feb)

Objectives

- [X] Understand how to export table data
- [X] Understand how to import data, especially CSV files
- [X] Know how to delete tables
- [X] Understand unique constraints (PRIMARY KEY)
- [X] Remaining DML commands: UPDATE TABLE
- [X] Remaining DDL commands: ALTER TABLE
- [X] NULL values and how to enforce them

SQLite

Delete tables

- `DROP TABLE [table];`
- PRACTICE

Export table content only

- You already know how to `.dump` an SQLite database `.db`
- Sometimes, you only want to export the data only, without the .~schema~

- For this, you can use `.mode insert`, then redirect the `.output` and `SELECT` the data for the output file
- PRACTICE

Export a CSV file

- First export your file as a CSV file
- Change `.mode`, `SELECT`, change `.output`, `SELECT` again
- Now you have a CSV file to import
- PRACTICE

Import a CSV file

- `.import [csv file] [table]`
- See help for `.import` on the SQLite shell
- If you have an error, how can you fix it?
- PRACTICE

Unique constraints

- Unique constraints: `PRIMARY KEY`
- What happens if you have defined a column `id` as `PRIMARY KEY`?
- Try to `INSERT` the same `VALUE` in a table that already has one
- Use `.schema [table]` to see your table definitions
- PRACTICE

UPDATE TABLE (DML)

- `UPDATE` table values with `UPDATE [table] SET [col]=[new];`
- If you're not careful you update entries that don't need it
- How can you specify a row and leave others unchanged?
- Use a filter with `WHERE` to specify the row!
- PRACTICE

ALTER TABLE (DDL)

- SQLite has limited capabilities to alter tables
- You can change the name, or you can add a column, that's it
- `ALTER TABLE [table] RENAME TO [newname];`
- Add a column with `ALTER TABLE [table] ADD COLUMN [name] TEXT;`
- PRACTICE

NULL values

- `NULL` values are special, too
- Check out the `.nullvalue` setting via `.show`
- Set it to `[NULL]` (default is "")
- `SELECT` all columns of the table you just added a column to
- Add a new column and label it `NOT NULL`
- PRACTICE

Next

- Interactive notebook (Org) to practice all of this
- More realistic database (`food.sql`) to practice
- DataCamp

SQL notebooks - w8s12 (1-Mar)

News

- DataCamp course changes - Intermediate SQL, 2 projects
- To count, courses must be published in Schoology
- Mid-term grades: check if you want to submit late

Objectives

- [X] Review quiz 4-5 (I'm waiting for your comments/questions)
- [X] Interactive SQLite notebook tutorial
- [X] SELECT notebook 1
- [X] Lecture: SELECT with examples

SQLite Code Blocks in Org Mode

- Do you remember what Org-mode is?
- Do you remember what tangling and weaving means?
- See FAQs (GitHub)
- See Orgmode documentation "SQLite code blocks in Org-mode"

Download practice material

- Download from GDrive
  - `init.el` / `.emacs` Emacs init file
  - `start_nb.org` file
  - `SELECT_nb_1.org` file
- Download from GitHub: `foods.sql`
- Work throught the getting started tutorial
- Begin with the `SELECT` notebook

Midterm grades, SQLite cloud REPL - w8s13 (3-Mar)

Objectives

- [X] How to improve your mid-term grades (FAQ)
- [X] Learn how to access SQLite in the cloud (REPL)
- [X] Find the updated notebook materials in GDrive
- [ ] Review the entire SELECT pipeline (for SQLite)

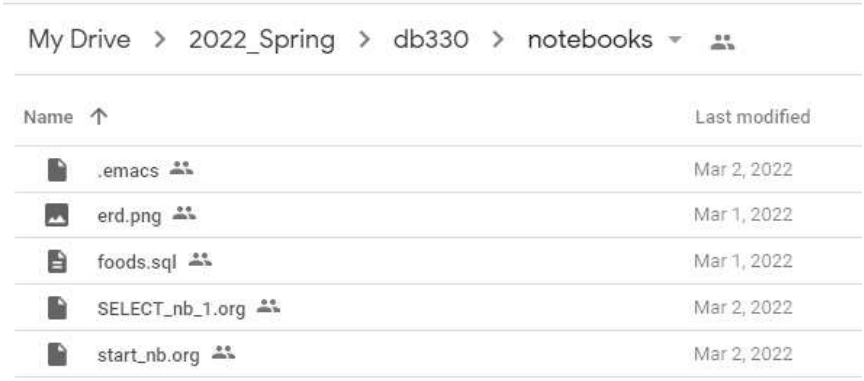


Figure 21: SELECT round-up

Mid-term grades improvement

- You can ask me personally and specifically, what to do to get your grades up
- There is no reason not to have at good grade in my class:

1. You can usually submit in-class assignments late
2. The deadlines of the DataCamp assignments are well known
3. The quizzes contain ample instructions and can be repeated
4. Class attendance + Whiteboard screenshots + GitHub info
5. You can always talk to me for personal support

- Hence, to improve your grade, do:
  - Submit in-class assignments if you could not attend class
  - Complete DataCamp assignments on time
  - Play the quizzes until you have 100% and read the feedback
  - Attend class + look at screenshots + files afterwards
  - Practice your skills whenever you can (e.g. DataCamp)
  - When you are attending in person, really attend
  - Ask me in or outside of class if anything is unclear
- These skills are related to successful studying, which in turn is related to success in life through traditional values: **discipline**, **duty**, and **diligence**. This doesn't have anything to do with computer science.

- What I'm going for in my classes is what I think computer scientists (and everyone!) need more than anything else:

  1. Critical thinking and analysis skills
  2. Troubleshooting skills
  3. Research skills

  This is nicely mirrored in this comment to the question "Why are computer science degrees so math intensive when the field doesn't seem to use much math at all?" on Quora.

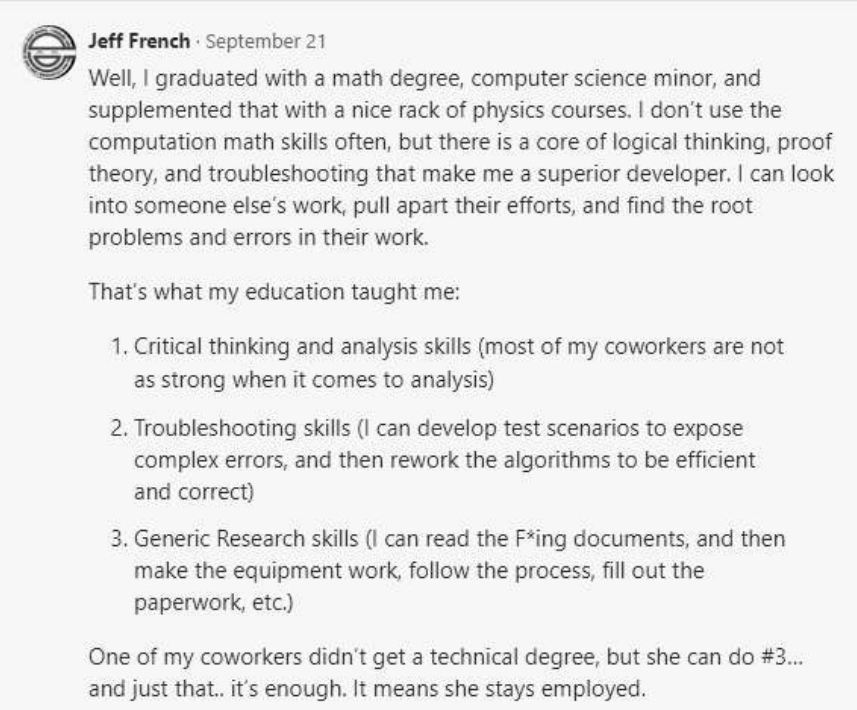


Figure 22: What's math got do to with computer science?

SQLite Read-Eval-Printing-Loop in the cloud

- Instead of a local terminal or application bundle like XAMPP, you can also use a cloud REPL to learn and practice SQL, SQLite
- `[X]` To begin login to your Google or GitHub account
- `[ ]`

  Register with Replit.com

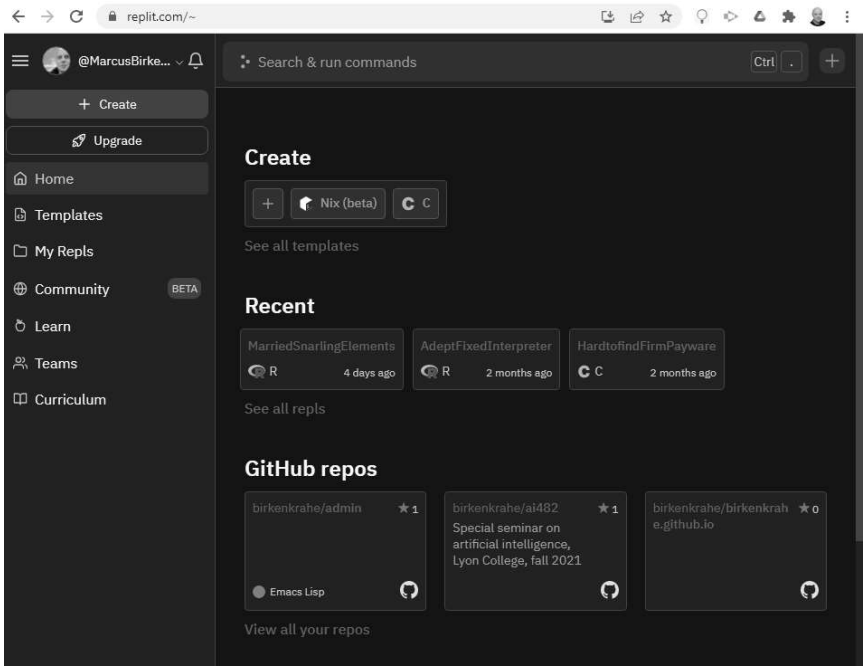


Figure 23: Replit.com startup screen

- Once registered, you can pick among many language templates
- You can create as many REPLs as you like on a free account

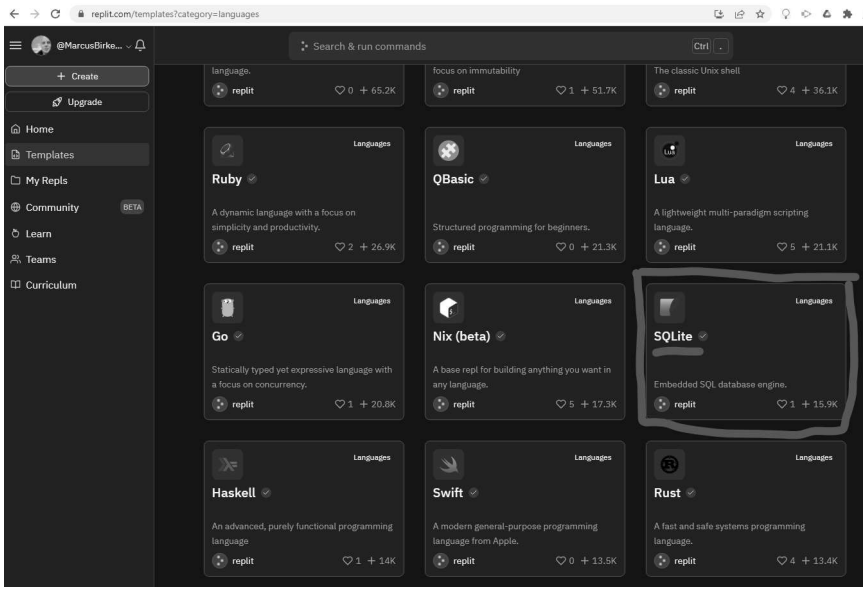


Figure 24: Replit.com language templates

- Find and pick SQLite! Your REPLs are now public and shareable.
- For example, to join the bash REPL shown below, use this link.

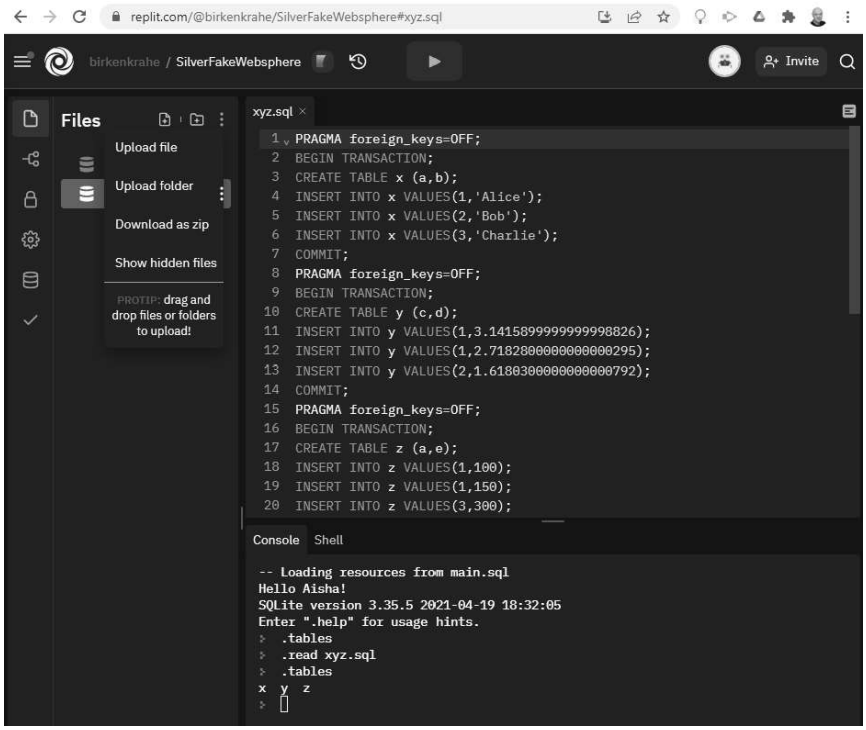


Figure 25: Create a titled, shareable repl

- You can now run SQLite commands and also access the `bash` shell.
- When you execute a command in the dashboard, it's the same thing as

```
sqlite3 < main.sql
```

- This is all made possible with a "Docker" container (with nix). More about that in the notes.
- We'll get back to this when we connect SQLite to other programs
- Today, just follow along using the REPL (or your local terminal, and if you don't have SQLite on your PC, download it now! Installation is trivial. All you need is to put the location of the `sqlite3.exe` file in your `PATH`. For Windows, you need `sqlite-tools-win32-x86-3380000.zip (1.87 MiB)`.)

SELECT lab 1: JOINs, WHERE - w9s14 (8-Mar)

Agenda

- [ ] Waking up in `AppData/Roaming`? Change your Emacs HOME now.
- [ ] Test preparations (Test on Thursday 17-Mar)
- [ ] Review Docker/container technology (notes)

Preparations for test 2 (Thu 17-Mar)

- Test 2 will only cover questions from quiz 4-6 + new questions.
- You can find quiz 4-6 with solutions + feedback as PDF (in /quiz)
- I will create an update of content Org files (in pdf/)

Objectives

- [ ] Review the entire SELECT pipeline (for SQLite)
- [ ] Review DataCamp (deadline 'sorting/grouping')

- [ ] Prepare for DataCamp project (sample project SQL)

SELECT round-up

- Download the notebook `SELECT_roundup.org` from GDrive (link in Schoology)
- Code along with me
- Finish the notebook on your own
- Work through this independently using `foods.sql` on Thursday

SELECT lab 1: ORDER BY, GROUP BY, LIMIT, OFFSET - w9s15 (10-Mar)

- [X] Finish SELECT workbook SELECT_roundup.org (GDrive)
- [X] Prepare DataCamp project: Introductory SQL project

Prepare DataCamp project

The upcoming DataCamp assignment is a "guided project", which means that you're told, in the usual DataCamp fashion, what to do, then enter and run the code. This time, however, you'll be working in a so-called Jupyter Notebook. That's a literate programming environment like Emacs Org-mode (just not nearly as flexible).

Before attempting the assigned project (deadline March 15), I recommend you look at the Introduction to Projects in DataCamp - it's very short and gives you the necessary background. You find it here. You can then head over to the SQL project on international debt statistics.

That's what I'm going to do!

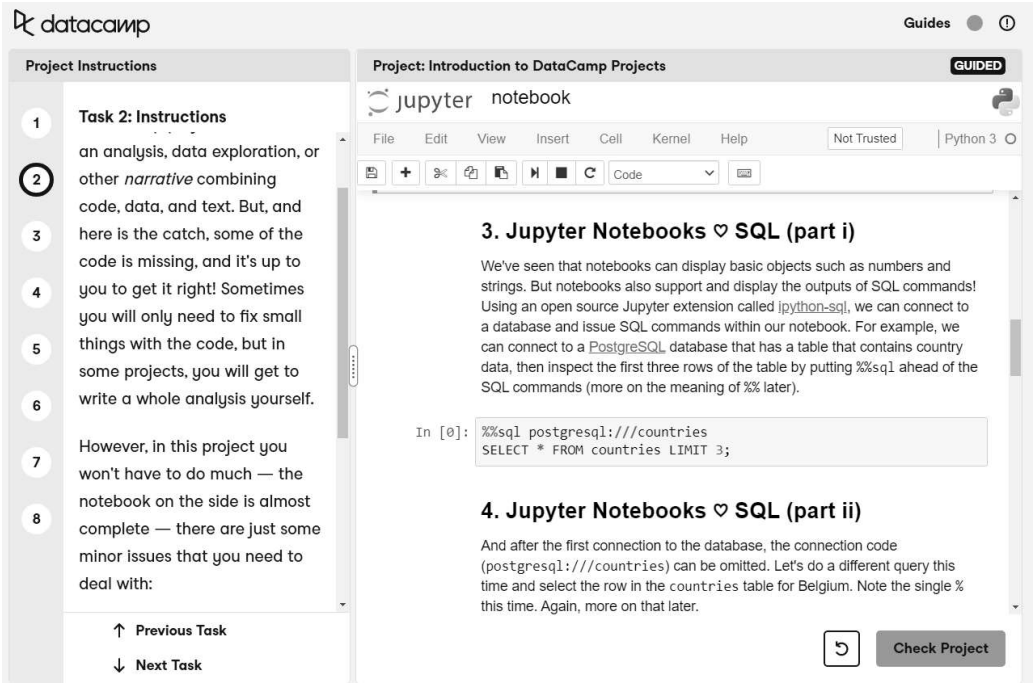


Figure 26: Screenshot of the Intro to Projects project

Quiz review, project review - w10s16 (15-Mar)

- Preview: European Soccer League database in SQLite format
- Review DataCamp project "Analyze International Debt Statistics"
- Review quiz questions Quiz 4 - 6 (PDF)

Preview Soccer database

I have tracked down the SQLite database version of the European Soccer league games. This is the dataset used in the upcoming DataCamp lessons. If you want to play with a much larger dataset - this is it: the SQLite db file is 300MB. To run it on the command line, use "sqlite3 soccer.sqlite" - then you're already there. Perfect opportunity to try some of the commands we've been learning on a larger canvas.

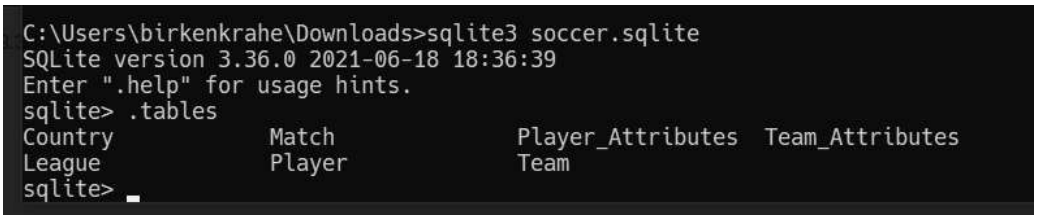


Figure 27: Loading soccer.sqlite on the CMD line

Introduction to DataCamp Projects

- Jupyter notebooks
- iPython
- PostgreSQL database
- Python pandas/DataFrame
- Object relation mapper
- Ponyorm.com
- SQLalchemy toolkit

Analyze International Debt Statistics

THE WORLD BANK's INTERNATIONAL DEBT DATA

- World Bank - what's the World Bank's mission?

  > "The World Bank Group has two goals, to end extreme poverty and promote shared prosperity in a sustainable way."

- What are the purposes of this project?

  - What is the total amount of debt that is owed by the countries listed in the dataset?
  - Which country owns the maximum amount of debt and what does that amount look like?
  - What is the average amount of debt owed by countries across different debt indicators?

- `[ ]`

  Attributes of interest:

  - total debt by country
  - maximum of debt by country
  - average debt by country

  ```
  SELECT debt, MAX(debt), AVG(debt)
    FROM [table]
    WHERE country='...'
  ```

- `[X]`

  Connecting to `international_debt` database

  What exactly is "PostgreSQL" (technically)?

- A (object-) relational database management system

What is the "cell magic" `%%sql` (technically)?

- meta data to enable query execution by the RDBMS

```
SELECT * FROM international_debt;
```

- `[X]`

  Output `LIMIT` to first 1-20 rows

  ```
  LIMIT 10 OFFSET 0;
  LIMIT 0, 10;
  LIMIT 10;
  ```

- `[ ]`

  Connecting to a database - what is generally involved?

  - Picking the database type (e.g. PostgreSQL, MySQL, SQLite)
  - Opening the database (e.g. `.open sqlite.db`)
  - Tables are loaded automatically

FINDING THE NUMBER OF DISTINCT COUNTRIES

- `[ ]`

  DISTINCT inside or outside of COUNT()?

  Can you explain the difference?

  ```
  SELECT DISTINCT
  COUNT(country_name) AS total_distinct_countries
  FROM international_debt;
  ```

```
SELECT
COUNT(DISTINCT country_name) AS total_distinct_countries
FROM international_debt;
```

In the first example, no column has been selected so the operator/function `DISTINCT` has nothing to act upon.

In the second example, the number of unique countries is selected - `DISTINCT country_name` throws away any rows with the same value in `country_name`.

FINDING OUT THE DISTINCT DEBT INDICATORS

- `[X]`

  ORDER BY

  - what does it order?
  - what's the default?
  - how can the default be changed?

    ORDER BY orders all values of a column in ascending (ASC) order unless the keyword DESC is used (descending order) after the column name.

TOTALING THE AMOUNT OF DEBT OWED BY THE COUNTRIES

- `[X]`

Aggregate function with 2nd argument: ROUND(column,precision)

- More about `ROUND`
- A 'million million" is a trillion, better: 10^12 (1e+12)

```
SELECT
ROUND(SUM(debt)/100000, 2) AS total_debt
FROM international_debt;
```

COUNTRY WITH THE HIGHEST DEBT

- Did you expect the answer?
- Check out this visualization (2021) - what's the difference?
- International debt statistics breakdown

```
SELECT country_name, sum(debt) AS total_debt
 FROM international_debt
 GROUP BY country_name
 ORDER BY total_debt DESC
 LIMIT 1;
```

AVERAGE AMOUNT OF DEBT ACROSS INDICATORS

- GROUP BY two columns - what does that mean?
- What happens if you take the 2nd argument away?

GROUP BY x,y means "put all rows with the same values for BOTH x and y in the same group".

```
SELECT
 indicator_code AS debt_indicator,
 indicator_name,
 AVG(debt) AS average_debt
 FROM international_debt
 GROUP BY debt_indicator, indicator_name
 ORDER BY average_debt DESC
 LIMIT 10;
```

System message:

```
* postgresql:///international_debt
(psycopg2.ProgrammingError) column "international_debt.indicator_name"
must appear in the GROUP BY clause or be used in an aggregate function
LINE 3:     indicator_name,
      ^
[SQL: 'SELECT \n    indicator_code AS debt_indicator,\n    indicator_name,\n    AVG(debt) AS average_debt\nFROM i
(Background on this error at: http://sqlalche.me/e/f405)
```

See SQLalchemy link

THE HIGHEST AMOUNT OF PRINCIPAL REPAYMENTS

- What's special about the filtering with WHERE?
- What's the subquery nesting depth allowed?

```
SELECT country_name, indicator_name
 FROM international_debt
 WHERE debt= (SELECT max(debt)
              FROM international_debt
              WHERE indicator_code='DT.AMT.DLXF.CD');
```

THE MOST COMMON DEBT INDICATOR

- ORDER BY with two arguments - is the order important?
- What happens if you turn the arguments around?

```
SELECT indicator_code, COUNT(indicator_code) AS indicator_count
FROM international_debt
GROUP BY indicator_code
ORDER BY indicator_count DESC, indicator_code DESC
LIMIT 20;
```

OTHER VIABLE DEBT ISSUES AND CONCLUSION

In [193]:
```
%%sql
SELECT country_name, max(debt) AS maximum_debt
FROM international_debt
GROUP BY country_name
ORDER BY maximum_debt DESC
LIMIT 10;
```

```
 * postgresql:///international_debt
10 rows affected.
```

Out[193]:

country_name	maximum_debt
China	96218620835.699996948
Brazil	90041840304.100006104
Russian Federation	66589761833.5
Turkey	51555031005.800003052
South Asia	48756295898.199996948
Least developed countries: UN classification	40160766261.599998474
IDA only	34531188113.199996948
India	31923507000.799999237
Indonesia	30916112653.799999237
Kazakhstan	27482093686.400001526

Figure 28: DataCamp project last code/output cell

DISCUSSION

- [X] What's left out in this analysis of debt?
- [X] Has the exercise changed your point of view?
- [X]

  Should analysis like this change your point of view?

  > "This debt is the sum of different debts owed by a country across several categories. This will help to understand more about the country in terms of its socio-economic scenarios."
  >
  > "we took a look at debt owed by countries across the globe. We extracted a few summary statistics from the data and unraveled some interesting facts and figures. We also validated our findings to make sure the investigations are correct."

Relational database thinking - w11s18 (29-Mar)

- [x] Group exercise: relational database thinking
- [x] Object-relational mapping in editor.ponyorm.com
- [x] Presentation of your ideas

Group exercise

Most importantly: DO NOT FEEL RESTRAINED BY REALITY (including your own technical abilities). Areas of interest could include:

- Video games
- Sports
- Alien civilizations
- Weather on Mars
- Tank movements in the Ukraine-Russia war
- Ecological data
- Immigration
- Kardashians
- The 100 richest people on Earth
- etc.

- [ ] Go into groups of 2-3 people - generate 1 solution per group
- [ ]

  Which database content you'd find personally interesting

  1. [ ] which area of knowledge or activity?
  2. [ ] which entities are worth measuring and recording?
  3. [ ] which variables would you measure?
  4. [ ] which data types do these variables come in?
  5. [ ] how "big" are your data?
  6. [ ] which queries would be interesting?
  7. [ ] write a couple of queries in pseudocode!
  8. [ ] which graphical outcomes would you like to see?
  9. [ ] is your information tabular/relational or not? If it is, how many tables do you think there are and what are their relationships?
  10. [ ] Put your results into the ponyorm editor (see below)

  Get ready to present your results as a group!

Example: Seinfeld foods

1. **Area:** of interest: Food science, advertising, marketing
2. **Entities:** Food types, foods, shows, actors, humor, US food sales data, viewer data
3. **Variables:** Food type, food name, show episode, actor name, humor index, food type sales figures, viewer age
4. **Data types**: text, integer, double
5. **Big data**: yes if sales and viewers included, certainly when doing this in real time
6. **Queries:** "how many more burgers were sold after Kramer ate a burger in episode 5?"
7. **Pseudocode:** SELECT count(*) FROM food_type, actors, episodes WHERE episode = 5 GROUP BY burgers
8. **Graphs:**
9. **DB type:** relational (tables)
10. See foods database in ponyorm

Ponyorm editor

- Go to editor.ponyorm.com and sign up for the free plan
- Click on "Create diagram" (blue button)
- Name your diagram, describe it briefly, set to PUBLIC
- Create entities with "New Entity"
- Add attributes (= variables) with "Add attribute"
- Add relationships with "Add relation"

- Share the public link with everyone when presenting

Presentations

- See GDrive for results (ponyorm designs)

SELECT lab 2 - foods database (31-Mar)

- [ ] FYI: DataCamp deadlines adjusted
- [ ] Work through the notebook at your own pace (GDrive)

Entity Relationship Diagrams (ERD) (5-Apr/7-Apr)

Emacs tip: hide emphasis markers in Emacs

- [ ]

  Add the string `(setq-default org-hide-emphasis-markers t)` to your `$HOME/.emacs` file so that you don't see the emphasis markers in Org-mode.

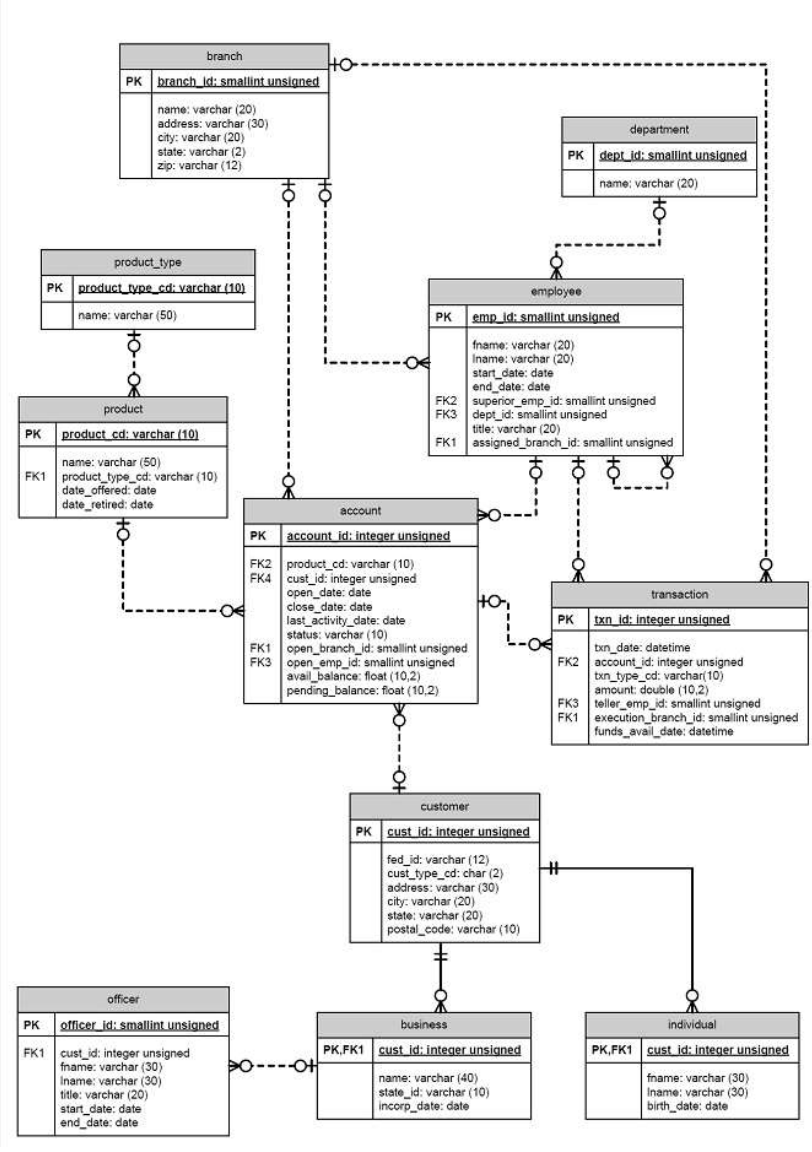


Figure 29: ERD of a bank (Source: Beaulieu, 2008)

ERD

- [X] ERD in Crowfoot notation and Chen notation
- [X] Cardinality/Multiplicity
- [X] Drawing a correct relational DB ER Diagram
- [ ] Primary Key, Foreign Key, Composite Key
- [ ] Bridge tables and data types
- [X] ERD Practice

See also: Lucidchart Tutorial (2017)

ERD and the Relational Model (12-Apr)

- [X] Review ERD practice exercise (Schoology)
- [X] ERD II: keys & bridge tables (practice/ in GitHub)
- [X] ERD practice exercise (GDrive)

ERD practice / Intermediate SQL lab: CASE (14-Apr)

- [X]

  No class on Tue 19-Apr and Thu 21-Apr.

  > On **Tuesday, April 19**, and on **Thursday, April 21** at **1 pm** in **Derby Science room 16**, instead of class, you are invited to join the teaching presentations of two candidates for a math professor position at Lyon. **This will count as your attendance on Tue/Thu.** The presentation will take about 45 minutes. It will be followed by Q&A. At the start of the week, you'll get a notebook from me (30-60 minutes completion time) to practice your intermediate SQL skills in line with the DataCamp lessons. Thank you!

- [X]

  Dropped last two DataCamp assignments - one mandatory assignment remains (deadline 21 April), and a bonus assignment (project). If you finished "Intermediate SQL" => extra credit.

  > FYI: I have removed the last couple of DataCamp assignments since the total number of assignments was already 12 for this course, and that's enough - also we won't have enough time to go over the material in class. Instead, we'll take a look at a couple of important topics like NoSQL/unstructured data, and the last project (bonus assignment). However, if you already finished "Intermediate SQL", let me know and I give you extra credit.

- [X] Finish the ERD exercise from last Tuesday if you have not already done so. Ask for my help if you can't do it!
- [X] Everybody else, please work on Intermediate SQL notebook `intermediate.org` in GDrive. Finish this at home if you need to. (Solutions as PDF in the `pdf/` directory in GitHub).

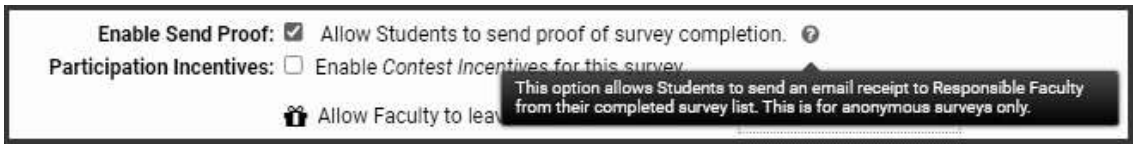
Review Quizzes before Test 3 (26-April)

Course evaluations - open until May 5

- [X] Made a lot of changes after the last evaluations.
- [X] Support the development of this course
- [X]

  Help me and Lyon College decide if I stay or if I go

  Extra credit for evaluating! - Don't forget to click the box.



… or send me an email that you did it (I'll verify with admin)

Plans for the last weeks of term (Schoology update)

- On **Tuesday, April 26**, we will review the 2 quizzes for *test 3* (scheduled for online submission Thursday 28-April 6 pm am to Saturday 30-April 8 am).
- On **Thursday, April 28**, we will review NoSQL and graph databases (this is also the deadline for the video-based assignment).
- On our last meeting, on **Tuesday May 3**, we will talk about the final exam (scheduled Thursday 5-May 6 pm to Saturday 7-May 8 am), summarize what we've achieved together, and wrap up with leftovers (1-2 items to be chosen from a large menu including Tableau, R and SQLite, Count.io and other SQL dashboards, the final DataCamp bonus project, and much more).
- You will get a detailed list with curated links and other sources from me for the DIY continuation of your database theory and applications journey (@ end of May), and a *bonus video* if the muse cooperates and if your evaluations are mostly positive :-)

Figure 31: Two German soccer players shouting "finals!"

Review quiz 7-8

- [x] We will review quiz 9 (SQL subqueries) on Thursday
- [x] Quiz 7: Entity Relationship Diagrams
- [x] Quiz 8: Intermediate SQL 1

Review quiz 9 (28-April)

What's so difficult about computer science/data science?

- [x] What's difficult for you?
- [x] What's difficult for me?
- [x] What can you do about it?
- [x] Will this ever change?

"How to solve it"

- [ ] Practical remedy against getting overwhelmed by too much detail: develop powerful heuristic strategies
- [ ] Use Euclid's math (2,500 years of proven problems) - or read a lot of blogs on the same problem and identify the patterns
- [ ]

  Go with Euclid: book recommendation for the holidays! "How to solve it" by George Pólya (1990 - see also application example).

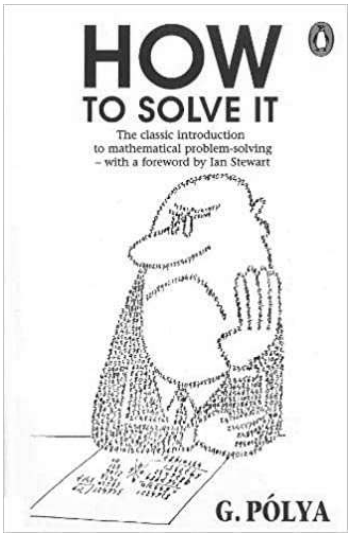


Figure 32: Perennial classic: Pólya's HOW TO SOLVE IT

- [ ]

  Traditional teaching vs. modern problem solving

  > »The traditional math professor of the popular legend is absentminded. He usually appears in public with a lost umbrella in each hand. He prefers to face the blackboard and to turn his back on the class. He writes `a`, he says `b`, he means `c`, but it should be `d`. Some of his sayings are handed down from generation to generation:
  >
  > *"In order to solve this differential equation you look at it till a solution **occurs** to you."*
  >
  > *"This principle is so perfectly general that no particular **application** of it is possible."*
  >
  > *"Geometry is the art of correct **reasoning** on incorrect figures."*
  >
  > *"My method to overcome a difficulty is to **go round** it."*
  >
  > *"What is the difference between method and device? A method is a **device** which you use twice."*

Intermediate SQL notebook

- [X] My personal notebook for the course is on GDrive/GitHub
- [X] Took me about 1 full work day to complete
- [X] Highly recommend this way of working with online courses
- [X] You should hold yourself accountable for time you spend on X
- [X] You should be able to recall an online course like a book ($\frac{5}{=}$)

Deadlines

- [ ] **Test 3 from Thu-28-Apr 6 pm to Sat-30-Apr 8 am**
  - Open for unlimited play from now
  - Quizzes will be disabled before the test
- [ ] **Final exam from Thu-5-May 6 pm to Sat-7-May 8 am**
  - Random draw from all questions + minor modifications

- Quizzes and tests will be disabled before the exam

Beyond SQL, Final Exam Review (3-May)

Final Exam Schedule (Schoology)

COURSE NO	MEETS	TOPIC	TIME	ROOM
CSC 330	TR 1.00p	Databases	Sat 7-May 1.00p - 3.00p	Lyon 104
			Fri 6-May 1.00p - 3.00p	Derby 209
			Fri 6-May 3.30p - 5.30p	Derby 209
			Mon 9-May 1.00p - 3.00p	Derby 209
			Mon 9-May 3.30p - 5.30p	Derby 209

The final exam will take place **in room 104 of the Lyon building** (our usual classroom) on **Saturday, May 7 from 1.00 pm to 3.00 pm** (see exam schedule).

The exam will consist of 40 questions drawn at random from the pool of quiz and test questions. Completing the various quizzes and tests until you've reached 100%, and revisiting your past tests should enable you to pass this exam with flying colors!

Please reach out to me if you have difficulties taking the exam at this time. I'm going to offer a limited number of seats to take this exam on Friday May 6 or on Monday May 9. Graduating seniors must complete the exam no later than May 8.

Figure 33: Black and white movie GIF

Beyond SQL

MongoDB playground

- Video summary & analysis
  - Entities = Collections + Tables

    - Documents = Schema-less (Rows)
    - Good for: missing values (NAs)
    - Advantage: flexibility for expansion
    - Graph sorting algorithms
    - Programming with dictionaries (Python), lists (R)
    - Less merging, faster searches, duplicate data
    - Video presents possibilities - for OLAP, storage, performance and scaling are only two aspects. More important to providers than to (analytical) users.
    - Additional issue: hardware dependency - which technology makes best use of existing and potential performance boosts like
        - parallelization (code issue = RDBMS + query)
        - quantum computing (e.g. cybersecurity applications)
        - distributed data (e.g. many small data centers)
    - Meta issue: the way in which data are organized impact our ability to discern and display facts about the world!
    - Real world dependencies: Google had to go back to SQL services because most real world web applications rely on SQL databases
    - NoSQL is not a new paradigm. There hasn't been a new paradigm in database technology for 70 years… (something like OOP)
    - NoSQL is actually kind of an old paradigm because collections resemble spreadsheets more than relational databases
- Playing with MongoDB
    - If you want to try out MongoDB (popular NoSQL database) see here
    - Docker container image for download or online playground
    - Regular SQL (MySQL), MongoDB, Cypher (graph database language)
    - NoSQL databases require much more effort to do OLAP
    - To try:
        - `db.getCollectionNames();` (returns collection names)
        - `db.products.dataSize();` (returns collection size)

Name	Description
`db.collection.aggregate()`	Provides access to the aggregation pipeline.
`db.collection.bulkWrite()`	Provides bulk write operation functionality.
`db.collection.count()`	Wraps `count` to return a count of the number of documents in a collection or a view.
`db.collection.countDocuments()`	Wraps the `$group` aggregation stage with a `$sum` expression to return a count of the number of documents in a collection or a view.
`db.collection.createIndex()`	Builds an index on a collection.

Figure 34: MongoDB reference commands (mongodb.com)

RDBMS for data science: DuckDB

- Data scientists (and OLAP[6] people) don't like/use RDBMS:
    - They're really slow (subqueries?)
    - Queries are hard to optimize (1960s programming)
    - Little parallelism
    - No out-of-memory computation (like interpreted language)
    - No transactional storage
- Re-think RDBMS from the benefit backwards, not from the available technology forward!
- Some **Germans** (!) have created a state-of-the-art analytical DBMS
    - It's free, open source and bypasses expensive consultants
    - It's fast and comfortable
    - It's written in C++ and supports SQL (PostgreSQL syntax)
    - APIs for C, C++, Python, CLI, Python, R, Java, Julia, Go…
    - It's in-process like SQLite (!)
    - No external dependencies (like "Tidyverse") - raw power

    - Apache Arrow table and Apache Parquet format support[7]
    - Focus on R (visualization and analysis > table munging)
- They called it DuckDB because ducks are cute (and tasty)
- Online playground available
- Intro talk by Hannes Mühleisen at NYhackr (Mar 1, 2022)

DataCamp webinar: work of a Data Engineer

- Check out the recent webinar + slides
- Interesting industry-relevant take on data work
- "Custom SQL" and management tools
    - Versioning (= Git as in GitHub)
    - Data Warehousing (OLAP, Apache tools)
    - Dashboarding (Tableau, R Shiny)

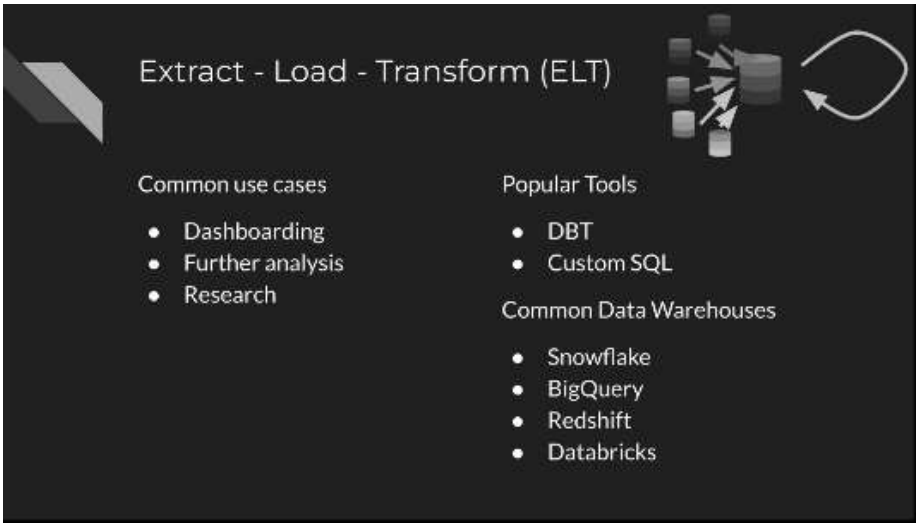


Figure 35: ETL data engineering (Source: DataCamp)

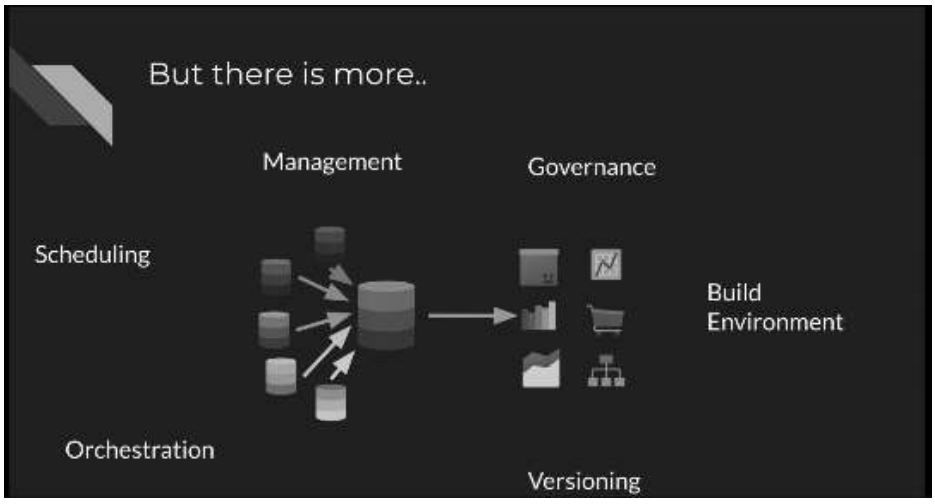


Figure 36: ETL data engineering (Source: DataCamp)

Final Exam Review

- [X] **Any outstanding work must be finished by Monday May 9**
- [X] Discuss questions not in the last block (Q7-Q9/Test 3)
- [X] Will curate & modify questions here and there before Sat

Evaluation

- [ ] Changes that I would make:
  1. More non-SQL: MongoDB demo, DuckDB, GraphDB, XML
  2. Group project: building, connecting, running a DB
  3. Fewer DataCamp assignments, more DIY notebooks
  4. Homework should include reading articles/book chapters
- [ ]

  Please take 10 minutes to complete the course evaluation now

Final exam (Sat 7 May 1-3 pm)

COURSE NO	MEETS	TOPIC	TIME	ROOM
CSC 330	TR 1.00p	Databases	Sat 7-May 1.00p - 3.00p	Lyon 104
			Fri 6-May 1.00p - 3.00p	Derby 209
			Fri 6-May 3.30p - 5.30p	Derby 209
			Mon 9-May 1.00p - 3.00p	Derby 209
			Mon 9-May 3.30p - 5.30p	Derby 209

The final exam will take place **in room 104 of the Lyon building** (our usual classroom) on **Saturday, May 7 from 1.00 pm to 3.00 pm** (see exam schedule).

The exam will consist of 40 questions drawn at random from the pool of quiz and test questions. Completing the various quizzes and tests until you've reached 100%, and revisiting your past tests should enable you to pass this exam with flying colors!

Please reach out to me if you have difficulties taking the exam at this time. I'm going to offer a limited number of seats to take this exam on Friday May 6 or on Monday May 9. Graduating seniors must complete the exam no later than May 8.

References

- Beaulieu (2008). Learning SQL. O'Reilly.
- Lemahieu et al (2021). Principles of Database Management. Univ of Cambridge Press. URL: pdbmbook.com.
- Lucidchart (2017). Entity Relationship Diagram (ERD) Tutorial Part 1 [video]. URL: youtu.be/QpdhBUYk7Kk.
- Lucidchart (2017). Entity Relationship Diagram (ERD) Tutorial Part 2 [video]. URL: youtu.be/-CuY5ADwn24.
- TutorialCup (n.d.) System Catalog [website]. URL: www.tutorialcup.com.

Footnotes:

[1] For installation on your PC, see these instructions (PDF). The installation is simple: download the ZIP file, unpack it, and set the PATH variable so that sqlite can be found.

[2] In Lyon 104 (Computer lab), the PATH variable is unfortunately stored in the personal user's app data so that I could not set it properly for your account (you can do this on your own PC easily). However, I seem to have found a solution for this (see course FAQ).

[3] It is not true that a "table is like a spreadsheet". Spreadsheets contain active fields for computation. Org-mode tables are actually spreadsheet-enabled:

```
|         2 |
|         5 |
|         7 |
|-----------|
| 4.6666667 |
#+TBLFM: @5$1=vmean(@1..@4)
```

[4] On Windows, Emacs Org-mode cannot handle

[5] Think about it: when you read a book (on paper, not on the screen), you work your way through much more like with a notebook: e.g. you read around the text, you flip through pages to find stuff you forgot, you look way ahead to see what else is there etc. This is arguably much more effective than just watching something, though less effective (I find) than trying to comprehend through re-creating the material (which also double-checks it - never trust anyone blindly :-).

[6] OLAP = OnLine Analytic Processing, part of Business Intelligence or BI, including relational database querying, data and process mining

[7] Arrow is for efficient tabular formatting of data, Parquet for efficient storage, where "efficiency" relates to the ability to analyze very large amounts of data fast.

Author: Marcus Birkenkrahe
Created: 2022-05-03 Tue 15:20