TDA357/DIT621 – Databases

Lecture 1A – Course Introduction

Jonas Duregård

Who am I?

- Jonas Duregård
- Senior lecturer at the Computing Science division
- Fifth time I am responsible for this course
- Took this course myself back in 2006, when I studied in the CS Program (Datavetenskapligt program)

Who are you?

- A *lot* of people, as it turns out. A mix of:
  - Computer Engineering (TIDAL)
  - Industrial Engineering and Management (TKIEK)
  - Software Engineering (TKITE)
  - Biomedical Engineering, Interaction Design, Data Science (MPxyz)
  - Computers Science and other GU-students
  - Possibly other people that no one even told me about (let me know you exist!)
- You have learned (and not completely forgotten) set theory and such
- You have some experience of programming in Java or similar
- A course in data structures is not required, but may be helpful

Questions

- Can be asked at any time. I'll try to remember to pause and ask occasionally.

Why databases?

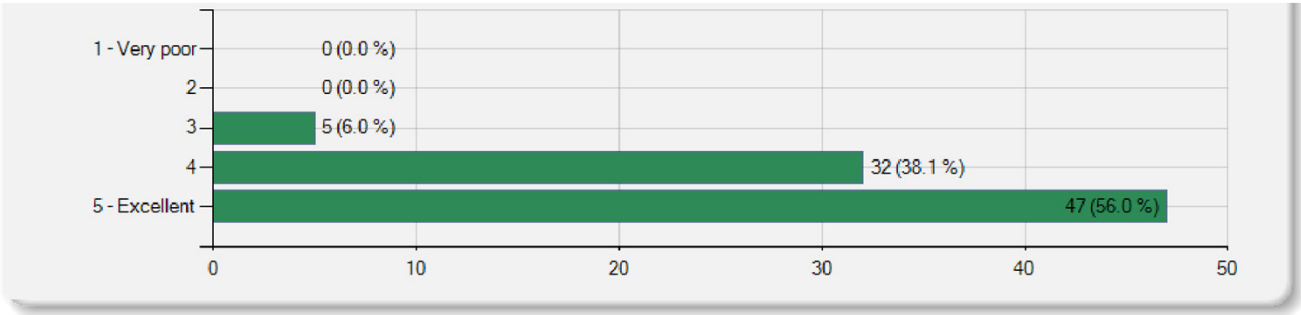
Let's get motivational

Why a whole course on databases?

- Short answer: Databases are everywhere
- Long answer: Databases are used in
  - WWW (Every page you visit involves several database operations)
  - Finance (has driven the development of databases since the 60's)
  - Industry: Production control, Test data, Inventories, Sales, …
  - Research: Sensor data, Biological data, Demographical data…
  - It would be a challenge to go a single day of your life without causing a lot of databases updates

More reasons to (keep) taking databases

- Previous instances of the course have been phenomenally well received



	Mean	Median
What is your overall impression of the course?	4.50	5.00

- Select comments:
  ”it is probably one of the best courses so far on Chalmers”
  ”One of the better lecturers I've ever had”
  ”Best lab assignments I've ever had in a course”
  ”Great PowerPoints with sufficiently dank memes!”

Wow, that was motivating – now what is a DB?

- Wide sense: Any collection of data that can be accessed digitally and is:
  - Structured – Data is stored in efficient structures
  - Persistent – Data is not lost without deliberate action
  - Mutable – data can be added/deleted/modified
- Slightly more useful: A database is a collection of data managed by a specialized software called a Database Management System (DBMS)

Some popular DBMS's:
Oracle, MS SQL Server, PostgreSQL, MySQL, …

Why not use a file system?

- File systems are structured, persistent and mutable
- … but very inefficient and 'bulky' to work directly with

Modern DBMS

- Handle persistent data
- Give efficient access to huge amounts of data
- Guarantees *integrity constraints* on data
- Handles transactions and concurrent access to data

Relational databases

- Basically, a bunch of tables with columns and rows
- Can also be viewed as mathematical relations, if you're into that
- Requires significant design-work
- SQL is a standardized language for manipulating relational databases
  - Common language supported by lots of DBMS
  - Create, manipulate and query databases
  - Arguably one of the most used computer languages in existence
  - Fancy people pronounce it "Sequel"
    - Exactly one person (Matthías Páll) pronounces it "Squirrel"

Other database models

- The relational data model is so prevalent, other approaches are commonly referred to as NoSQL-databases
- Semi-structured hierarchical models (XML, JSON, …)
- Key-value stores (Oracle NoSQL, Riak, …)
  - Easily distributed across multiple computers/data centers
  - Very simplistic data model (maps and lists)
- Usually NoSQL-databases are easier to design, sometimes more efficient, but also more limited when it comes to integrity constraints

Course Objectives

You will learn how to

- Design a database
- Construct a database from a schema
- Use a database through queries and updates
- Use a database from an external application

Course Objectives – Design

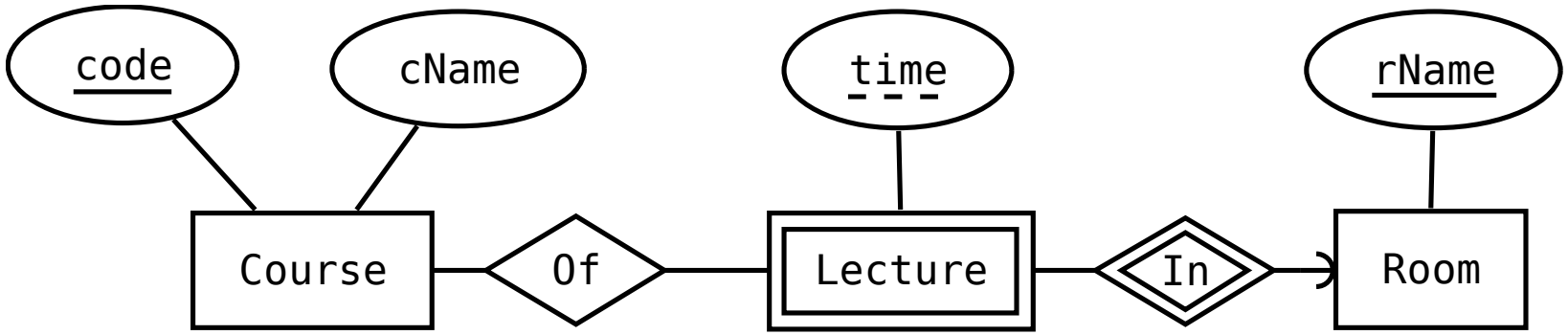
- When the course is through, you should
  - Given a domain, know how to design a database that correctly models the *domain* and its constraints

> ”We want a database that we can use for scheduling courses and lectures. This is how it’s supposed to work: …”

Course Objectives – Design

- Construct Entity-Relationship digagrams (ER)
- Determine functional dependencies (FD's)
- Compute normal forms

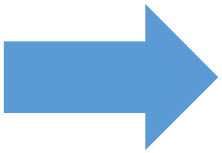
```
code → cName
code, time → rName
rName, time → code
```



Course Objectives – Construction

- When the course is over, you should
- Given a database schema with related constraints, implement the database in a relational DBMS

```
Courses(code, cname)
Rooms(rname)
Lectures(room, time, course)
    room   -> Rooms.rname
    course -> Courses.code
```



```
CREATE TABLE Courses(
  code CHAR(6) PRIMARY KEY,
  cname TEXT NOT NULL
);
CREATE TABLE Rooms (
  rname VARCHAR(100) PRIMARY KEY
);
CREATE TABLE Lectures (
  room VARCHAR(100) NOT NULL
    REFERENCES Rooms,
  course CHAR(6) NOT NULL
    REFERENCES Courses,
  time TIMESTAMP,
  PRIMARY KEY (room, time)
);
```

Course Objectives – Usage

- When the course is through, you should
  - Know how to query a database for relevant data using SQL
  - Know how to modify the contents of a database using SQL

```
SELECT time, room
  FROM Lectures
  WHERE course = 'TDA357';
```

```
INSERT INTO Rooms VALUES ('GD');
```

Course Objectives - Application

- When the course is through, you should
  - Know how to connect to and use a database from external applications

```
Class.forName("org.postgresql.Driver");
Properties props = new Properties();
props.setProperty("user", user);
props.setProperty("password", pwd);
conn = DriverManager.getConnection(db, props);
```

Switchin' it up

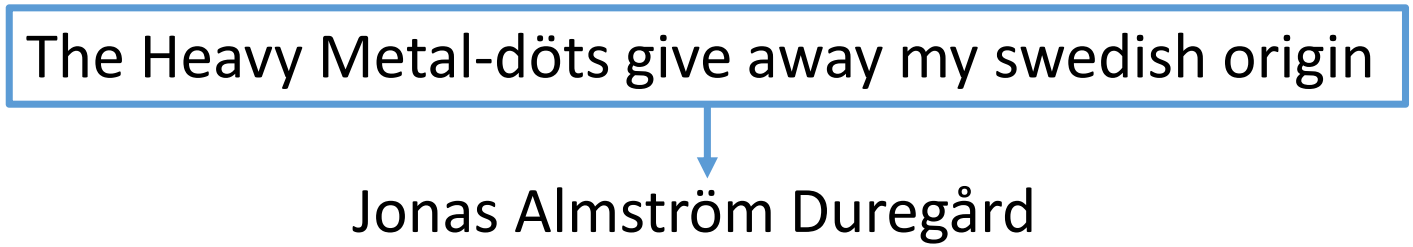
- Traditionally this course is taught in the order you perform the tasks: design > construction > usage > application
- Problem: High level design concepts are difficult to learn and appreciate before you know about usage
- Instead, we will start with the gritty details (SQL), and then learn the abstract concepts on top of that:
  Construction > Usage > Design > (more usage/construction) > Applications

Course dates

- Scheduled lectures, lab sessions etc. up until December 17
- Exam is on January 11, afternoon (register for it now!)
  - There is a re-exam in March (regular exam period) and one in August

Course language

- … is English in case you hadn't noticed
- Feel free to ask me questions in Swedish and I will answer in English



Course organization

- 1-3 lectures per week
- An exercise sessions each week in weeks 1-6 of the course
  - Practice exam-like questions on the material from this week's lectures
- About 5 lab sessions per week (starting next week)
  - Opportunity to ask questions about the lab
  - No single session is mandatory, but you will have to attend at least one to demonstrate your solution (usually near the end of the course)
- As always, the schedule is available in Timeedit

Online/on-campus teaching

- Lectures are given on-campus, with recordings on a best effort basis.
- Exercises are given on-campus, but also streamed via Zoom.
- Each lab-group is assigned one on-campus lab session per week
  - The person grading your labs will be on those sessions
  - Not mandatory, but a good opportunity for feedback and help
  - Online help is given in parallel, using Slack to request help and Zoom to facilitate it

Weekly themes

- Each week has one or two themes for the lectures, and then an exercise on the same topic at the end of the week
- This week's theme is SQL. On Friday there will be an exercise where I show you how to work in SQL, might be very useful for the assignment
- Future themes:
  - **Week 2**: Design using ER
  - **Week 3**: Design using functional dependencies and normal forms
    Deadline of Assignment Task 1 on Tuesday
  - **Week 4**: Constraints and triggers + Database connectivity and JDBC
    Deadline of Assignment Task 2 on Friday
  - **Week 5**: Semi-structured data: JSON and NoSQL
  - **Week 6**: Transactions + Relational algebra
    Deadline of Assignment Task 3 on Tuesday
  - **Week 7:** Repetition and history of databases, Final deadline on Friday

Examination

The course has two parts:

- A lab assignment (graded pass/fail)
- A written exam (graded fail/3/4/5 or fail/G/VG)

- Both must be passed to pass the course
- The grade of the exam will be your course grade

Lab organization

- Assignments are carried out in groups of two
  - NO! – you are not an exception to this ☺ (Unless you are…)
  - Never more than two people per group
- The assignment is divided into five tasks:
  - Tasks 0-3 are handed in and graded
  - Task 4 is assessed on any lab session (don't wait until the last one)
  - You may and should proceed with the next task before the first is accepted
- All questions about groups etc. should be directed to my Head of Assignments Matti. Contact information on the web page.
- Questions about the assignment text can be posted to the Slack (but do not post any part of your solution!)

Task 0

- You need to register in the lab submission system Fire, form groups and submit a submission this week, containing only your preferred time for weekly on-campus lab session (follow instructions carefully)
- Use the Slack if you want to find a lab partner (there's a channel for it)
- When registering, make sure to provide your real name the same way it is shown in Ladok/Canvas
- On-campus lab session times are scheduled first-come-first-served! Submit your preference early if you have strong time constraints.
  - Submission of Task 0 opens 13:00 today, let's crash Fire!

Lab assignment overview

Goal: Construct a "student portal" application in Java+SQL

- Part I: Starting with a domain description and a draft schema, implement the schema in a database and write queries (views) for common operations
- Part 2: Use systematic design methods to add a few features, and find and eliminate flaws in the original schema
- Part 3: Create triggers to further improve the database
- Part 4: Connect to your database from a simple Java Application, use JSON to send your data to a web client
  - Python might be an option if there's significant interest

Feedback

- You should get a response to your submission within 3 working days from submission
  - This is an ambitious goal, so do not be to upset if we fail around deadlines
- If you are 'shotgunning' lots of poor submission, do not expect lots of useful feedback. We also reserve the right not to grade submissions at all if resubmissions do not show clear improvement.
- If you know your solution does not work when you submit it, always say so in your submission
- If you do not follow the submission instructions exactly, expect your submission to be summarily rejected without feedback (other than a polite request to follow the instructions)

About cheating

The assignment is to be carried in groups of two.

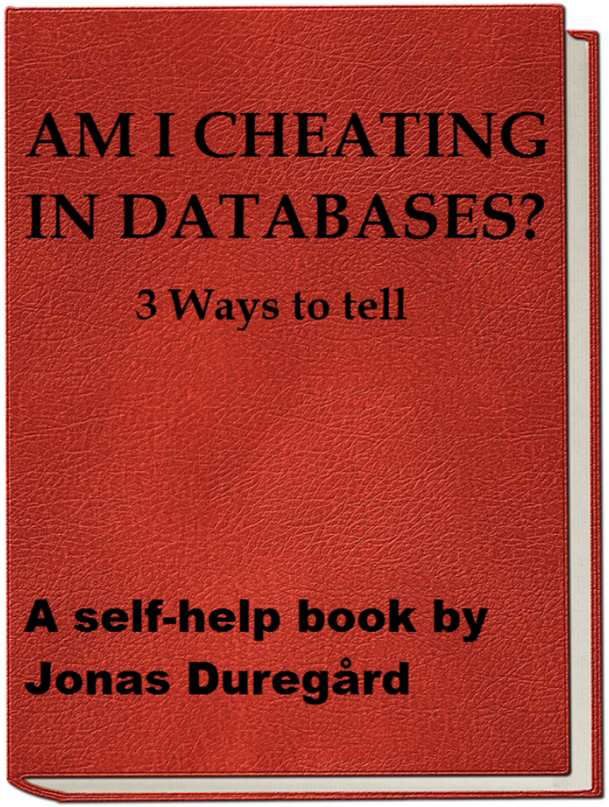
In the lab session, you will work close to other groups on the same tasks. Helping each other is allowed within certain limits.

It's easy to accidentally cross the line into cheating

- Generally, talking about the assignment is not cheating
- If you show your code to other students, you are cheating
- If you look at code from other students, you are cheating

If we catch you cheating, we are obligated to report it to a disciplinary committe

Also: Do not put your solution in a public repository!



Literature

- There is a manuscript by Jyrki Nummenmaa and Aarne Ranta currently called "Databases in 137 Pages" (link on the course page)
- There is also a book called Database Systems: The Complete Book by Garcia-Molina, Ullman and Widom.
  - This is a 1200 page volume. Reading it from cover to cover may not be the most productive use of your time…
- Also, the slides from these lectures are published online

- Freely available online sources should be all you need for the course

Web resources

- Course web page has everything (including these slides):

https://chalmers.instructure.com/courses/20962

(Since you are reading this, you have likely already found the page)

- Slack is used for a lot of communication, including asking questions about the lab (without posting any parts of solutions!) and requesting help during lab sessions
- Fire is used for submitting labs 🔥

Tools

- PostgreSQL (a.k.a. Postgres) is the DBMS we will be using in the course
  - Free stuff! Postgres is open-source software
  - Works on most operating systems
  - Available on Chalmers computers
  - postgresql.org
- Dia is the recommended editor for ER-diagrams
  - More free stuff! Dia is also open source!
  - Available on Chalmers computers (I hope)
  - dia-installer.de
  - Alternatively, you can have a look at diagrams.net
- You may want to have a text editor that supports SQL syntax highlighting

Teaching staff

- Examiner: Jonas Duregård (me!)
- A small army of Teaching Assistants:
  - Matthías Páll (it's the squirrel guy!)
  - William Bodin
  - Elias Ersson
  - Clara Salberg
  - Deaa Khankan
  - Albin Otterhäll
  - Carl Agrell

Any questions on course organization?