>>

COMP3311 Course Overview

- Why Study Databases?
- Databases: Important Themes
- What is Data? What is a Database?
- Studying Databases in CSE
- Syllabus Overview
- Your Background
- Teaching/Learning
- Lectures
- Labs and Tutes
- Prac Work and Tute Exercises
- Questions/Feedback/Issues
- Assignments
- Quizzes
- Exam
- Supplementary Assessment Policy
- Assessment Summary
- Textbook (options)
- Database Management Systems
- Further Reading Material
- Home Computing
- Course Schedule
- Overview of the Databases Field
- Database Application Development
- Database System Architecture
- Data Modelling
- SQL vs Relational Model
- ER-to-SQL Mapping

^ >>

COMP3311 Database Systems

Lecturer: *Raymond Wong* (cs3311@cse.unsw.edu.au)

Admin: *Nicole Lam* (cs3311@cse.unsw.edu.au)

Dominic Wong (cs3311@cse.unsw.edu.au)

Web Site: http://webcms3.cse.unsw.edu.au/COMP3311/21T1/
or http://www.cse.unsw.edu.au/~cs3311/

<< ^ >>

❖ Why Study Databases?

Every significant computer application involves Large Data.

This needs to be:

- stored (typically on a disk device)
- manipulated (efficiently, usefully)
- shared (by many users, concurrently)
- transmitted (all around the Internet)

Green stuff handled by databases; blue by networks.

Challenges in building effective databases: efficiency, security, scalability, maintainability, availability, integration, new media types (e.g., music), ...

<< ^ >>

❖ Databases: Important Themes

The field of databases deals with:

- data ... representing application scenarios
- relationships ... amongst data items
- constraints ... on data and relationships
- redundancy ... one source for each data item
- data manipulation ... declarative, procedural
- transactions ... multiple actions, atomic effect
- concurrency ... multiple users sharing data
- scale ... massive amounts of data

<< ^ >>

❖ What is Data? What is a Database?

According to the Elmasri/Navathe textbook ...

- Data = known recorded facts, with implicit meaning
  - e.g. a student's name, a product id, a person's address or birthday
- Database = collection of related data, satisfying constraints
  - e.g. a student *is enrolled in* a course, a product *is sold at* a store
- DBMS = database management system
  - software to manage data, control access, enforce constraints
- RDBMS = relational database management system
  - e.g. PostgreSQL, SQLite, Oracle, SQL Server, MySQL, ...

<< ^ >>

❖ Studying Databases in CSE

COMP3311 introduces foundations & technology of databases

- skills: how to build database-backed applications
- theory: how do you know that what you built was good

After COMP3311 you can go on to study ...

- COMP9313: managing Big Data (deal with a huge amount of data)
- COMP9315: how to build relational DBMSs (write your own PostgreSQL)
- COMP9318: techniques for data mining (discovering patterns in DB)
- COMP9319: Web data compression and search (dealing large amount of Web data)
- COMP6714: information retrieval, web search (dealing with text data)
- COMP9321: data services (making data available via a network)

<< ^ >>

❖ Syllabus Overview

Core syllabus ...

- Data modelling and database design
  - ER model, ODL, ER-to-relational
  - Relational model (design theory, algebra)
- Database application development
  - SQL, views, stored procedures, triggers, aggregates
  - SQLite: `sqlite3` (an SQL shell)
  - PostgreSQL: `psql` (an SQL shell), `PLpgSQL` (procedural),
  - Programming language access to databases (Python, ORMs)

The brown stuff is not covered in lectures and is not examinable

<< ^ >>

❖ Syllabus Overview (cont)

More syllabus ...

- Database management systems (DBMSs)
  - DBMS architecture: query processing, index structures
  - Transaction processing: transactions, concurrency control, recovery
- Future of Databases
  - Limitations of RDBMS's, potential future technologies

Blue and green stuff is covered only briefly, and is not examinable

To learn more about the green stuff, take COMP9313, COMP9319...

To learn more about the blue stuff, take COMP9315, ...

<< ^ >>

❖ Your Background

We assume that you ...

- have experience with procedural programming
- have some background in data structures
- hopefully, have some knowledge of Python

You might have acquired this background in

- COMP1511, COMP1531, COMP2521

If you don't know Python, look at some online tutorials soon.

e.g. https://www.python.org/about/gettingstarted/

<< ^ >>

❖ Teaching/Learning

Stuff that is available for you:

- Slides: summarize all syllabus topics
- Course Notes: contain extended versions of the slides (some are not examinable)
- Textbooks: describe most syllabus topics in detail
- Pre-recorded videos: elaborate all syllabus topics
- Live lectures (overview/problem-solving sessions): provide topic overview / work through examples
- Tutorials (written ex): design/concept/theory/written questions
- Labs (prac ex): hands-on prac work
- Assignments: more detailed practical exercises
- Quizzes: periodic progress check

All online. If you want on-campus, wait for COMP3311 21T3.

<< ^ >>

❖ Teaching/Learning (cont)

On the course website, you can:

- find out the latest course news
  (important annoucements will also be emailed)
- view the topic-based slides (videos on echo360)
- get details of tute/prac exercises
- get assignment specs/material
- do the quizzes
- get your quick questions answered (via the Forums)

URL: https://webcms3.cse.unsw.edu.au/COMP3311/21T1/

<< ^ >>

❖ Lectures

Similar to COMP3311 20T3:

- Pre-recorded videos
  - One topic per video
  - Available over the weekend in Echo360
- Live lecture sessions (on Blackboard Collaborate)
  - One 2-hr session per week (instead of two 1-hr sessions in 20T3)
  - Week 1: Mon 9-11am. Week 2-5,7-10: Wed 4-6pm
  - Overiew of the topics and/or problem-solving exercises
  - Recorded and available in Echo360

<< ^ >>

❖ Labs and Tutes

- Labs (2hrs; week 2,3,4,5)
  - Work on the prac exercises (ideally get them started before the lab)
  - Get help from your tutor on prac exercises (or tute ex / other COMP3311 related questions)
- Tutes (1hr; week 7,8,9,10)
  - Walk through solutions for difficult tute questions
  - Provide assignment feedback
  - Further explain selected database theories, if needed

<< ^ >>

❖ Prac Work and Tute Exercises

- Prac Work (week 1-5)
  - All prac exercises (week 1-5) are available now
  - You are expected to complete them every week accordingly
  - Get help from your lab tutor if you are stuck / have questions
  - Prac exercises equip you with the skills for the assignments
- Tute Exercises (week 2-5,7-10)
  - Tute ans to be released 1-2 weeks later
  - Difficult tute questions will be discussed in details in tutes (week 7-10)
  - You may ask your lab tutor if you are stuck with some tute questions (week 2-5)

<< ^ >>

❖ Questions/Feedback/Issues

You can communicate with us via:

- WebCMS forum (Quick questions / clarifications)
- Labs/tuts (Longer questions / interactions required)
- Live lectures (Lecture related questions)
- Email cs3311@cse.unsw.edu.au (Contact me / course admins: e.g., personal issues)
- Consultations (Anything else)

<< ^ >>

❖ Assignments

Two assignments, which are critical for learning

1. SQL/PLpgSQL, 20%, due end week 5
2. Python/SQL, 20%, due end week 9

All assignments are done individually, and ...

- submitted via `give` or Webcms3
- automarked (so you must follow specifcation exactly)
- plagiarism-checked (copying solutions ⇒ 0 mark for course)
- rent-a-coder monitored (buying solutions ⇒ exclusion)

<< ^ >>

❖ Quizzes

Eight quizzes, each worth 4 marks

- cover material in previous few weeks lectures
- aim to check your understanding of recent material
- done via Webcms3 in your own time
- primarily multiple-choice
- held in weeks 2, 3, 4, 5, 7, 8, 9, 10
- released Monday, due Friday 11:59pm
- can be submitted multiple times

8×4 = 32, which is mapped into a mark out of 10

Heavy penalties for late submission

<< ^ >>

❖ Exam

The Final Exam includes questions on ...

- SQL, PLpgSQL, (Python), design exercises, analyses
- 50% prac questions, 50% "written" questions (tentatively, to be confirmed later in the course)

Online, open-web exam during exam period

- content is what I'd put in a 3-hour in-lab exam
- exam is open for slightly longer than 3 hours
- can work on home machine (test on a cse machine before submission), or via `ssh`, or via `vlab`
- all questions typed in and submitted online (`give`)

A sample exam will be available on the course website in Week 10

<< ^ >>

❖ Supplementary Assessment Policy

Everyone gets **exactly one chance** to pass the Exam

If you attempt the Exam

- I assume that you are fit/healthy enough to take it
- no 2nd chance exams, even with a medical certificate

All Special Consideration requests:

- must document how you were affected
- must be submitted to UNSW (useful to email lecturer as well)

<< ^ >>

❖ Assessment Summary

Your final mark/grade will be determined as follows:

```
quizzes = mark for on-line quizzes    (out of 10)
ass1    = mark for assignment 1       (out of 20)
ass2    = mark for assignment 2       (out of 20)
exam    = mark for final exam         (out of 50)
okExam  = exam >= 22.5                (after scaling)

mark    = ass1 + ass2 + quizzes + exam

grade   = HD|DN|CR|PS  if mark >= 50 && okExam
        = FL           if mark <  50 && okExam
        = UF           if !okExam
```

<< ^ >>

❖ Textbook (options)

- Elmasri, Navathe
  Fundamentals of Database Systems (7th ed, 2016)
- Garcia-Molina, Ullman, Widom
  Database Systems: The Complete Book (2nd ed, 2008)
- Ramakrishan, Gehrke
  Database Management Systems (3rd ed, 2003)
- Silberschatz, Korth, Sudarshan
  Database System Concepts (7th ed, 2019)
- Kifer, Bernstein, Lewis
  Database Systems: Application-Oriented Approach (2nd ed, 2006)

Earlier editions of texts are ok

<< ^ >>

❖ Database Management Systems

Two DBMSs for this course:

- SQLite (open-source, free, no server needed)
- PostgreSQL (open-source, free, full-featured)

Comments on using a specific DBMS:

- the primary goal is to learn SQL (a standard)
- the specific DBMS is not especially important **
- but, each DBMS implements non-standard features
- we will use standard SQL as much as possible
- PG docs describe all deviations from standard

** Unless it seriously violates SQL standards ... I mean you, MySQL

<< ^ >>

❖ Further Reading Material

The on-line documentation and manuals provided with:

- SQLite are reasonably good
- PostgreSQL are very good
- Python are similarly comprehensive

Some comments on technology books:

- tend to be expensive and short-lived
- many provide just the manual, plus some examples
- generally, anything published by O'Reilly is useful

Aside: once you understand the concepts, the manual is sufficient

<< ^ >>

❖ Home Computing

Software versions that we'll be running this semester (TBC):

- PostgreSQL 12, SQLite 3.27, Python 3.7, psycopg2 2.8

If you install them at home:

- get versions "close to" these
- test all work at CSE machines before submitting

Alternative to installing at home:

- run them on the CSE servers (grieg) as you would in labs
- use `vlab` or `ssh` to log in to a CSE server from home

Details on starting sqlite3 and setting up a PostgreSQL server on grieg are in the first and second Prac Exercises. If you have any difficulties, get help from your lab tutors.

<< ^ >>

❖ Course Schedule

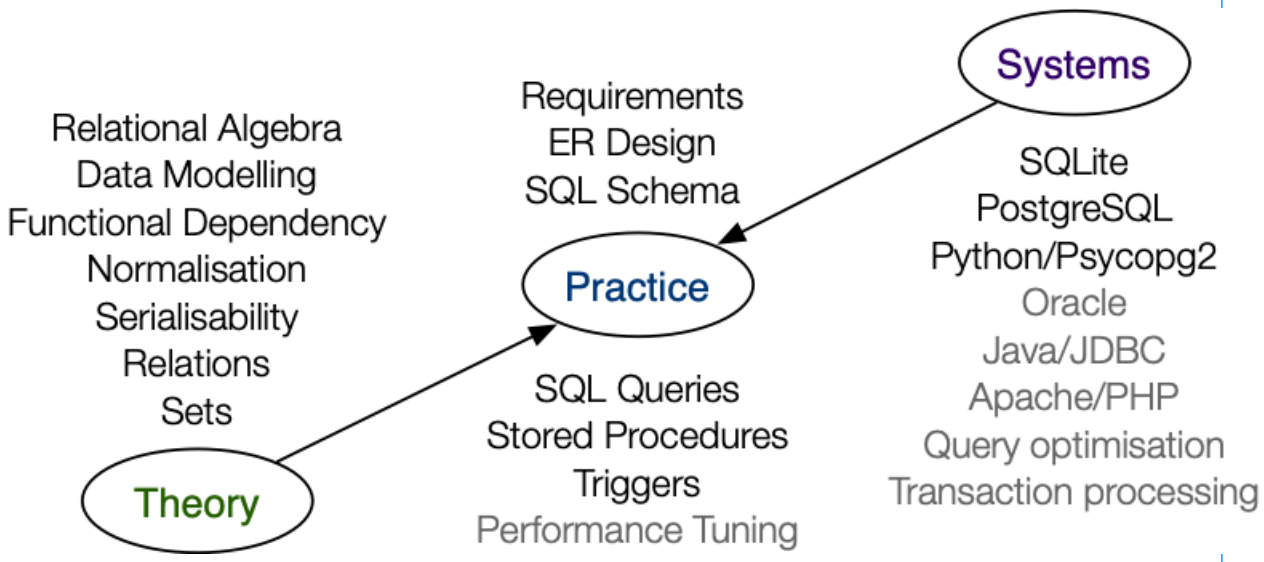
A tentative course schedule:

1. Requirements -> Data Model
2. Data Model -> Relational Schema (-> DBMS)
3. Database operations (i.e., SQL)
4. SQL
5. SQL
6. --
7. Python and SQL
8. Check for redundancy
9. Relational algebra and query execution
10. Transactions and concurrency control

Today, we will discuss what you are going to learn from the topic videos for week 1 and 2, i.e., database design

<< ^ >>

❖ Overview of the Databases Field



<< ^ >>

❖ Database Application Development

A variation on standard software engineering process:

1. analyse application requirements
2. develop a data model to meet these requirements
3. check data model for redundancy (using relational theory)
4. implement the data model as relational schema
5. define operations (transactions) on this model
6. implement operations via SQL and procedural PLs
7. construct a program interface to these operations
8. monitor performance and "tune" the schema/operations

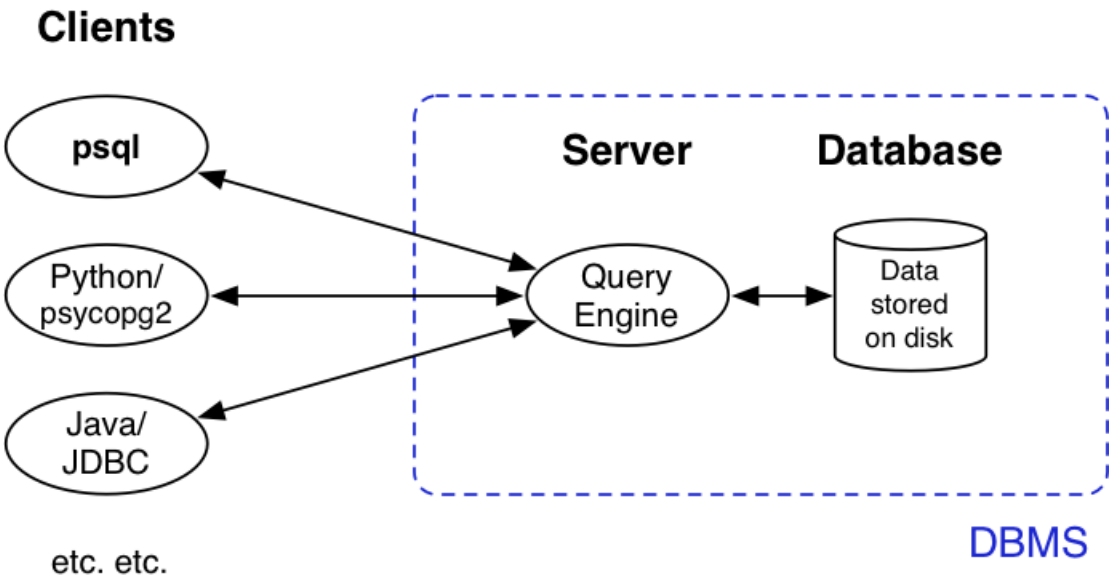
At some point, populate the database (may be via interface)

During the course, we consider these in the order 2, 4, 6, 7, 3

<< ^ >>

❖ Database System Architecture

The typical environment for a modern DBMS is:

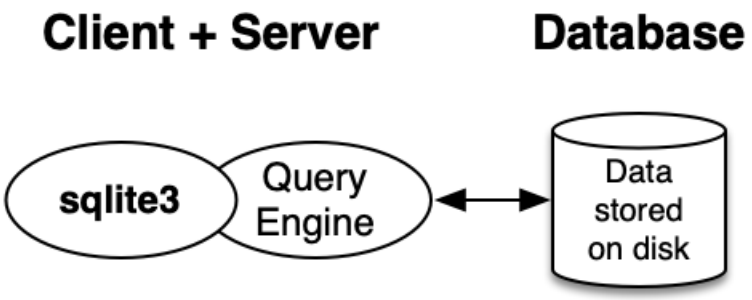


SQL queries and results travel along the client↔server links

<< ^ >>

❖ Database System Architecture (cont)

SQLite is not a client-server system:



Although it does have an API for use from programming languages.

<< ^ >>

❖ Data Modelling

Aims of data modelling:

- describe what information is contained in the database
  (e.g., entities: students, courses, accounts, branches, patients, ...)
- describe relationships between data items
  (e.g., John is enrolled in COMP3311, Tom's account is held at Coogee)
- describe constraints on data
  (e.g., 7-digit IDs, students can enrol in no more than 3 courses per term)

Data modelling is a design process

- converts requirements into a data model

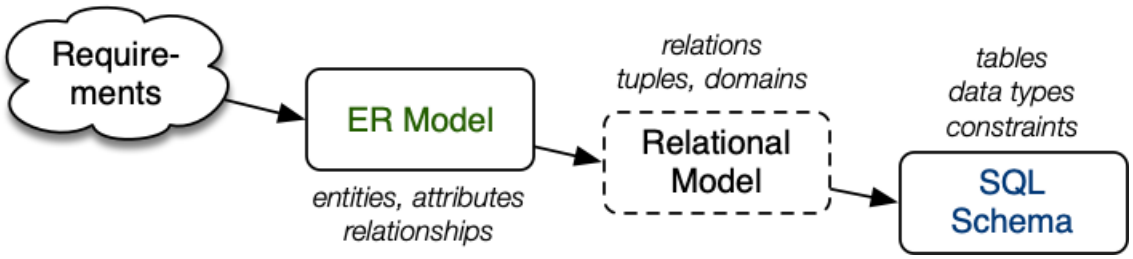
<< ^ >>

❖ Data Modelling (cont)

Kinds of data models:

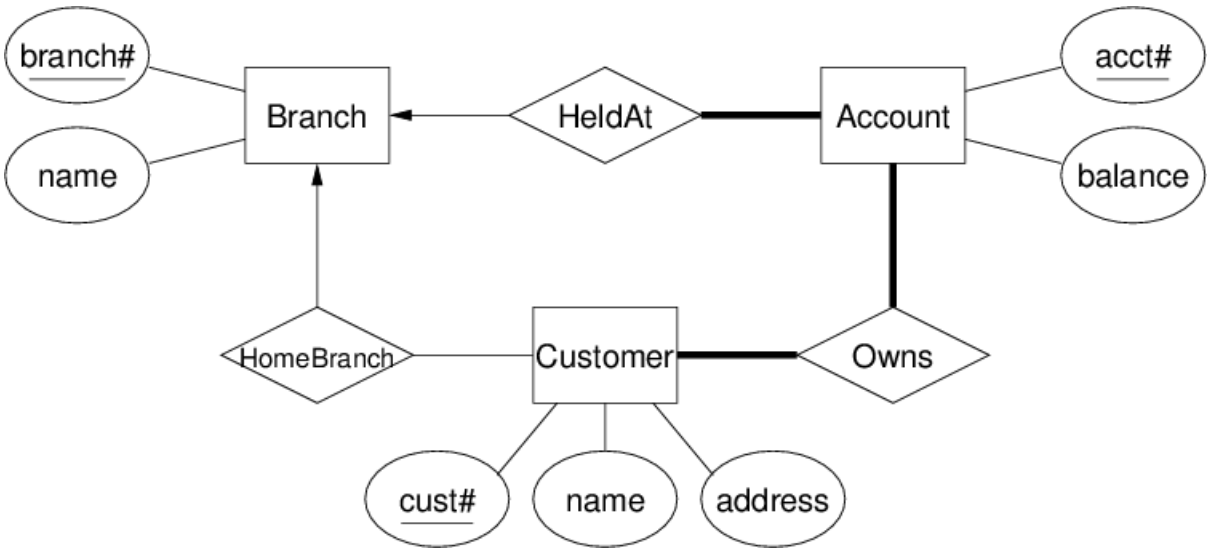
- logical: abstract, for conceptual design, e.g., ER
- physical: record-based, for implementation, e.g., relational, SQL

Strategy: design using abstract model; map to physical model



<< ^ >>

Entity-Relationship (ER) Diagrams **Example** ER diagram:



<< ^ >>

❖ SQL vs Relational Model

The relational model is a formal system for

- describing data (relations, tuples, attributes, domains, constraints)
- manipulating data (relational algebra ... covered elsewhere)

SQL is a "programming" language for

- describing data (tables, rows, fields, types, constraints)
- manipulating data (query language)

SQL extends the relational model in some ways (e.g bags vs sets of tuples)

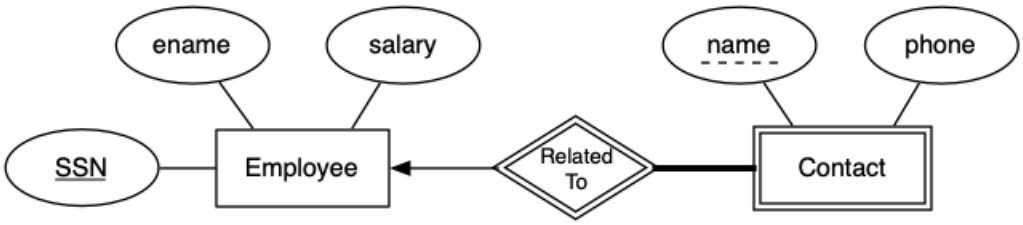
SQL omits some aspects of the relational model (e.g. general constraints)

<< ^

❖ ER-to-SQL Mapping

Example (Mapping Weak Entities):

ER Model



SQL Version

```
create table Employees (
    SSN      text primary key,
    ename    text,
    salary     currency
);
```

```
create table Contacts (
    relatedTo  text not null,  -- total participation
    name       text,           -- not null implied by PK
    phone      text not null,
    primary key (relatedTo, name),
    foreign key (relatedTo) references Employees (ssn)
);
```

Produced: 14 Feb 2021