COMP9311 Week 01 Lecture

COMP9311 Database Systems

Lecturer: Raymond Wong

Web Site: http://www.cse.unsw.edu.au/~cs9311/

(powered by WebCMS2 ... a PostgreSQL-backed web app)

Lecturer

2/50

Name:	Raymond Wong
Office:	K17-213
Phone:	93855932
Email:	wong@cse.unsw.edu.au
Research:	Mobile Data Management XML/Web Databases Information Retrieval Data Mining

Course Admin Supervisor

3/50

Name:	Mohammad Ebrahimi
Office:	K17 Level 2
Email:	mohammade@cse.unsw.edu.au

Why Study Databases?

4/50

Every interesting computer application has Big Data.

This needs to be:

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

Red stuff handled by databases; brown by networks.

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

Databases: Important Themes

5/50

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

Evolution of Databases

6/50

Network (60's)

- linked data items in file system, simple record structures
- programmatic access to data (no query languages)

Relational (70's - continuing)

- simple core concepts, useful theory, efficient implementations
- high-level query language: SQL
- ACID transactions, guaranteed consistency

OO (80's - 90's)

- attempted to overcome modelling deficiencies of relational
- overly complex implementations ... ultimately failed

XML (90's - continuing)

- similar rationale to OODB's, implementations still developing
- interesting query language with "structural" emphasis

Multimedia Retrieval (90's - continuing)

- similarity queries on music/images, still under research

noSQL (00's - continuing)

- distributed key-value stores for very large data (e.g. Google)
- access defined more procedurally than SQL (e.g. map/reduce)
- intended for non-ACID transaction scenarios, eventual consistency

ColumnStores (00's - continuing)

- twists relational model "sideways", still under research
- aims to provide efficient analytical processing

Databases in CSE

COMP9311 introduces foundations & technology of databases

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

After COMP9311 you can go on to study ...

- COMP9315: how to build relational DBMSs (write your own Oracle)
- COMP9318: techniques for data mining (discovering patterns in DB)
- COMP6714: information retrieval, web search (dealing with text data)
- COMP9319: web search and data compression (dealing searching compressed web data)
- COMP932x: service-oriented computing, which relies on DB background

---

Syllabus Overview

COMP9311 this semester will follow the style of COMP3311, and use the materials modified from Dr. John Shepherd.

- 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
  - PostgreSQL: psql (an SQL shell), PLpgSQL (procedural),
  - Introduction to programming language access to databases (PHP, ORMs)
  - Web interface technology: HTML (forms), PHP (scripting)

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

... Syllabus Overview

- Database management systems (DBMSs)
  - DB Administration: catalogues, access control, performance
  - DBMS architecture: client/server, file system, relational engine
  - Storage and indexing, data access operations
  - Query processing: translation, optimisation, evaluation
  - Transaction processing: transactions, concurrency control, recovery
- Future of Databases
  - Limitations of RDBMS's, potential future technologies

The green stuff is covered only briefly; details are in COMP9315.

Teaching/Learning

Stuff that's available for you:

- Texts: describe most syllabus topics in much detail
- Lectures: summarise all syllabus topics, with exercises

Things that you need to do:

- Theory exercises: tutorial-type questions
- Prac work: lab-class exercises (important)

- Assignments: extended practical exercises

... Teaching/Learning

11/50

Scheduled classes?

- there are allocated lab classes
- lab tutor will discuss difficult lab exercises
- lectures/web videos are useful

What to do if you have problems understanding stuff?

- ask a question in/after the lecture
- come to a consultation (check the course web page in case of time change)
- ask your lab tutor in the lab
- post to the web message board

... Teaching/Learning

12/50

On the course web site, you can:

- find out the latest course news/announcements
- view lecture slides
- get all of the information about theory/prac exercises
- find some useful questions/answers from the message board
- post questions to the message board (or even better, answer questions or share your views)

URL: http://www.cse.unsw.edu.au/~cs9311/

Assignments

13/50

Three assignments, which are critical for learning

1. data modelling, ER/SQL, due start week 4
2. queries/procedures, SQL/PLpgSQL, due start week 8
3. written assignment, due start week 12

All assignments are done individually. For programming assignments ...

- submitted via give
- automarked (so you must follow specifcation exactly)
- plagiarism-checked (copying solutions ⇒ 00FL for course)
- rent-a-coder monitored (buying solutions ⇒ expulsion)

Exam

14/50

There is a final written exam in November. Detail will be provided later.

Supplementary Assessment Policy

15/50

Everyone gets exactly one chance to pass the Exam.

If you attend 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 me as well)

Assessment Summary

16/50

Your final mark/grade will be determined as follows:

```
asgt1  = mark for assignment 1      (out of 25)
asgt2  = mark for assignment 2      (out of 50)
asgt3  = mark for assignment 3      (out of 25)
asgt   = asgt1 + asgt2 + asgt3      (out of 100)
exam   = mark for exam              (out of 100)

finalmark = 2 * asgt * exam / (asgt + exam)
```

Books

17/50

- Elmasri, Navathe (Textbook)
  Fundamentals of Database Systems (5th ed)
- Garcia-Molina, Ullman, Widom
  Database Systems: The Complete Book (2nd ed)
- Ramakrishan, Gehrke
  Database Management Systems (3rd ed)
- Silberschatz, Korth, Sudarshan
  Database System Concepts (5th ed)
- Kifer, Bernstein, Lewis
  Database Systems: Application-Oriented Approach (2nd ed)

Earlier editions of texts are ok (and cheaper, as are Kindle editions)

Database Management Systems

18/50

DBMS for prac work:

- PostgreSQL 9.0 (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
- an exception is PLpgSQL (but close to Oracle's PL/SQL)
- PG docs describe all deviations from standard

... Database Management Systems

Comments on PostgreSQL vs Oracle:

- Oracle is resource hungry (>800MB vs <200MB for PostgreSQL)
- PostgreSQL is a commercial-strength (ACID) RDBMS
  ... but, being open source, you can see how it works
- PostgreSQL has been object-relational longer than Oracle
  ... and its extensibility model is better than Oracle's
- PostgreSQL is more flexible than Oracle
  ... allows stored procedures via a range of programming languages

But note: PostgreSQL and Oracle have very close SQL and PL/SQL languages.

... Database Management Systems

Comments on PostgreSQL vs MySQL:

- both open source** and reasonably efficient
- most Web/DB developers use MySQL
- until v4/5, MySQL lacked many serious DB concepts
  - no transactions, foreign keys, subselects, views, procedures, ...
- MySQL's SQL often ignores SQL standards
- MySQL is hacked together from "imported components"
  - multiple storage engines (some still w/o transactions)

PostgreSQL is better engineered; MySQL is more popular.

** But Oracle now controls MySQL ⇒ open-source status unclear

Further Reading Material

Under the Documentation link on website:

- PostgreSQL has very good on-line documentation
- PHP has similarly comprehensive documentation

Some comments on PostgreSQL and PHP books:

- tend to be expensive and short-lived
- many provide just the manual, plus some examples
- generally, anything published by O'Reilly is useful
- make sure it deals with PHP5, PostgreSQL9, SQLite3

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

Home Computing

Software versions that we'll be running this semester:

- PostgreSQL 9.0, PHP 5.3, Apache 1|2?

If you install them at home:

- get versions "close to" these

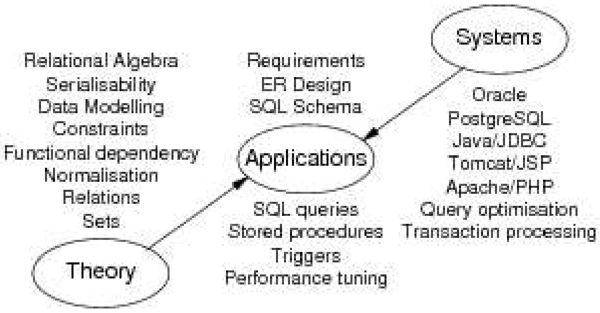
- test all work at CSE before submitting

Alternative to installing at home:

- run them on the CSE servers (grieg) as you would in labs
- use e.g. putty to log in to a CSE server from home
- PostgreSQL via putty ok, since command-line based
- to use Apache at CSE from home may require use of VPN

Overview of the Databases Field

23/50



(Will reveal the people behind these ideas via "DB Nerds" at start of lectures)

Database Application Development

24/50

A variation on standard software engineering process:

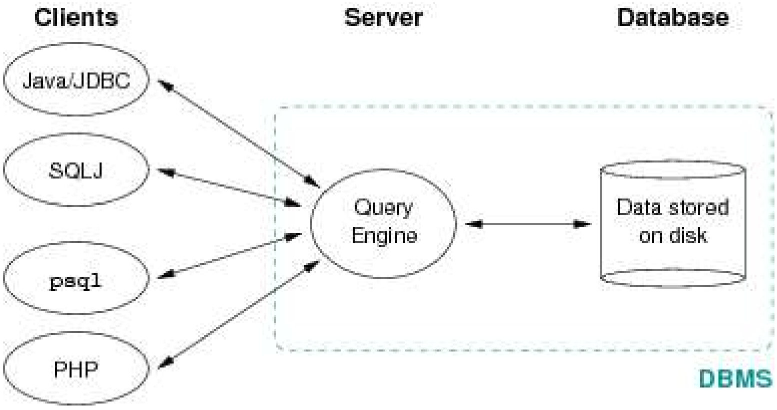
1. analyse application requirements
2. develop a data model to meet these requirements
3. define operations (transactions) on this model
4. implement the data model as relational schema
5. implement transactions via SQL and procedural PLs
6. construct a web interface to these transactions

At some point, populate the database (via interface?)

Database System Architecture

25/50

The typical environment for a modern DBMS is:



SQL queries and result tuples travel along the client↔server links.

Database System Languages

Requests to DBMS:

- queries, updates in data manipulation language
  (e.g. SQL)
- data structures, constraints in data definition language
  (e.g. SQL)
- create and drop databases, indexes, functions
  (e.g. PLpgSQL)

Results/effects from DBMS requests:

- tuples (typically, sets of tuples)
- changes to underlying data store

Data Modelling

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 COMP9311, Paul's account is held at Coogee)
- describe constraints on data
  (e.g. 7-digit IDs, students can enrol in no more than 30UC per semester)

Data modelling is a design process

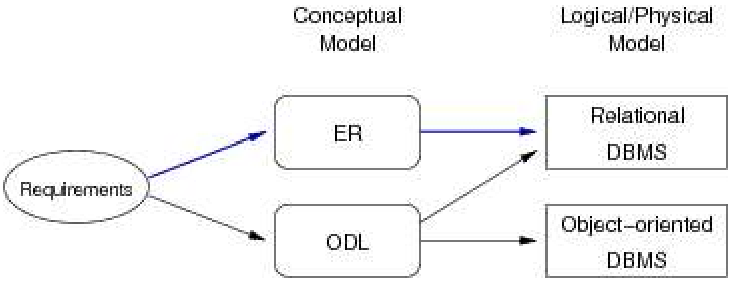
- converts requirements into a data model

... Data Modelling

Kinds of data models:

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

Strategy: design using abstract model; map to physical model



Some Design Ideas

Consider the following while we work through exercises:

- start simple ... evolve design as problem better understood
- identify objects (and their properties), then relationships
- most designs involve kinds (classes) of people
- keywords in requirements suggest data/relationships
(rule-of-thumb: nouns → data, verbs → relationships)
- don't confuse operations with relationships
(operation: he buys a book; relationship: the book is owned by him)
- consider all possible data, not just what's available

Exercise: GMail Data Model

31/50

Consider the Google Mail system.

Develop an informal data model for it by identifying:

- the data items involved (objects and their attributes)
- relationships between these data items
- constraints on the data and relationships

Exercise: Amazon Data Model

32/50

Consider the Amazon web site

Develop an informal data model for it by identifying:

- the data items involved (objects and their attributes)
- relationships between these data items
- constraints on the data and relationships

Quality of Designs

33/50

There is no single "best" design for a given application.

Most important aspects of a design (data model):

- correctness (satisfies requirements accurately)
- completeness (all reqs covered, all assumptions explicit)
- consistency (no contradictory statements)

Potential inadequacies in a design:

- omits information that needs to be included
- contains redundant information (⇒ inconsistency)
- leads to an inefficient implementation
- violates syntactic or semantic rules of data model

Entity-Relationship (ER) Model

Entity-Relationship Data Modelling

35/50

The world is viewed as a collection of inter-related entities.

ER has three major modelling constructs:

- attribute: data item describing a property of interest
- entity: collection of attributes describing object of interest
- relationship: association between entities (objects)

The ER model is not a standard, so many variations exist.

Lecture notes use notation from KSS and GUW books (simple)

Entity-Relationship (ER) Diagrams

36/50

ER diagrams are a graphical tool for data modelling.

An ER diagram consists of:

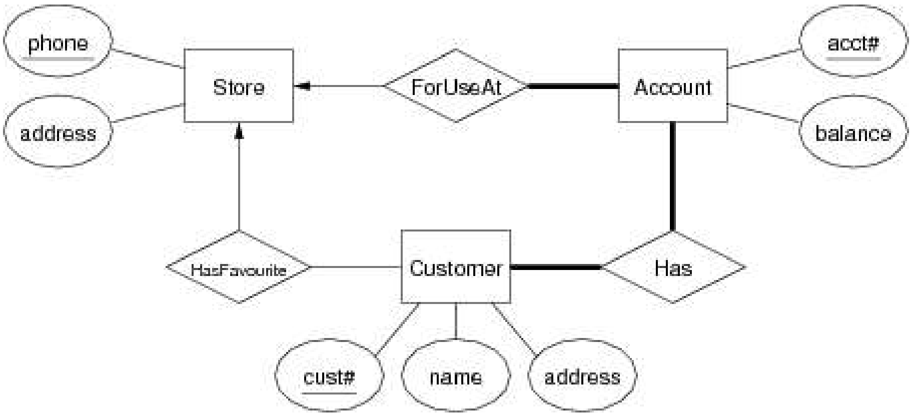
- a collection of entity set definitions
- a collection of relationship set definitions
- attributes associated with entity and relationship sets
- connections between entity and relationship sets

Terminology abuse: "entity" means "entity set" or "entity instance"?

... Entity-Relationship (ER) Diagrams

37/50

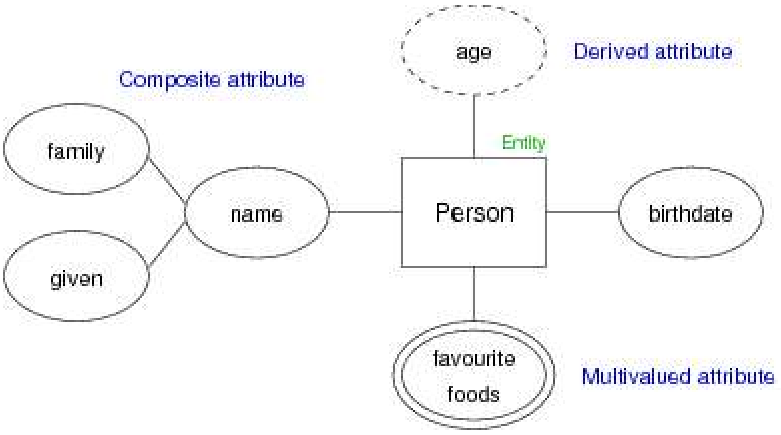
Example ER diagram:



... Entity-Relationship (ER) Diagrams

38/50

Example of attribute notations:



Entity Sets

39/50

An entity set can be viewed as either:

- a set of entities with the same set of attributes
  (extensional view of entity set)
- an abstract description of a class of entities
  (intensional view of entity set)

An entity may belong to more than one entity sets.

"Data" in a database ≅ collection of (extensional) entity sets.

Keys

40/50

Key (superkey): any set of attributes

- whose set of values are distinct over entity set
- natural (e.g. name+address+birthday) or artificial (e.g. SSN)

A candidate key is any superkey such that

- no proper subset of its attributes is also a superkey

A primary key:

- is one candidate key chosen by the database designer

Keys are indicated in ER diagrams by underlining.

Relationship Sets

41/50

Relationship: an association among several entities.

E.g. Customer(9876) is the owner of Account(12345)

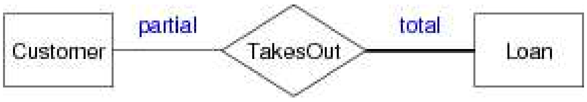
Relationship set: collection of relationships of the same type.

Degree = # entities involved in reln (in ER model, ⩾ 2)

Cardinality = # associated entities on each side of reln

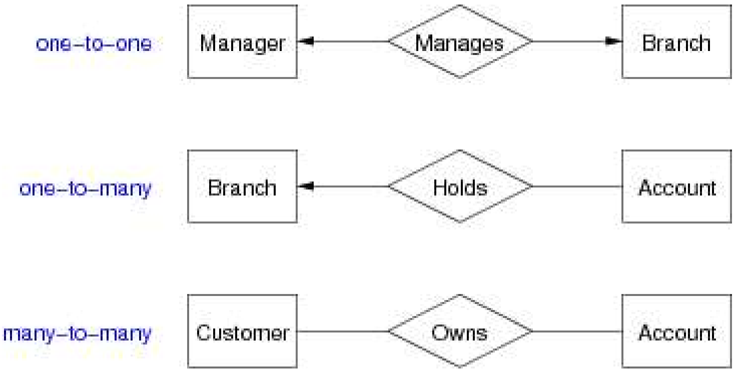
Participation = must every entity be in the reln

Example: relationship participation



... Relationship Sets

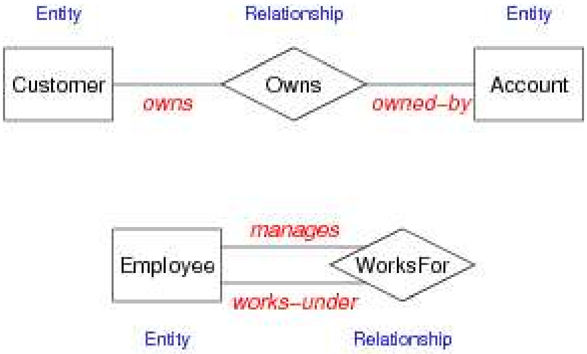
Examples: relationship cardinality



... Relationship Sets

The role of each entity in a relationship is usually implicit.

If ambiguity arises, can explicitly name the role.

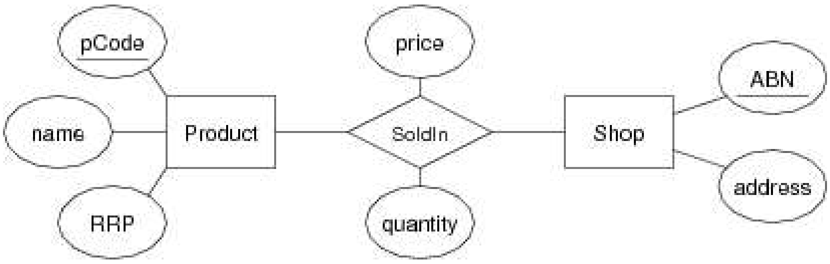


Role names become more important when developing SQL schemas.

... Relationship Sets

In some cases, a relationship needs associated attributes.

Example:



(Price and quantity are related to products in a particular shop)

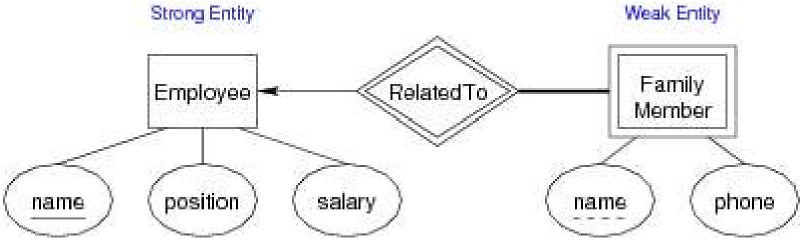
Weak Entity Sets

45/50

Weak entities

- exist only because of association with strong entities.
- have no key of their own; have a discriminator

Example:



Subclasses and Inheritance

46/50

A subclass of an entity set A is a set of entities:

- with all attributes of A, plus (usually) it own attributes
- that is involved in all of A's relationships, plus its own

Properties of subclasses:

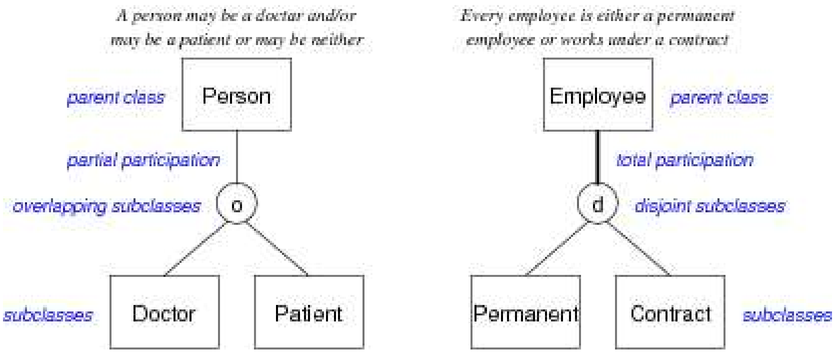
- overlapping or disjoint (can an entity be in multiple subclasses?)
- total or partial (does every entity have to also be in a subclass?)

Special case: entity has one subclass ("B is-a A" specialisation)

... Subclasses and Inheritance

47/50

Example:



Design Using the ER Model

48/50

ER model: simple, powerful set of data modelling tools.

Some considerations in designing ER models:

- should an "object" be represented by an attribute or entity?
- is a "concept" best expressed as an entity or relationship?
- should we use n-way rel^{n}ship or several 2-way rel^{n}ships?
- is an "object" a strong or weak entity? (usually strong)
- are there subclasses/superclasses within the entities?

Answers to above are worked out by thinking about the application domain.

... Design Using the ER Model

49/50

ER diagrams are typically too large to fit on a single screen.
(or a single sheet of paper, if printing)

One commonly used strategy:

- define entity sets separately, showing attributes
- combine entitities and relationships on a single diagram
  (but without entity attributes)
- if very large design, may use several linked diagrams

Exercise: Medical Information

50/50

Develop an ER design for the following scenario:

- Patients are identified by an SSN, and their names, addresses and ages must be recorded.
- Doctors are identified by an SSN. For each doctor, the name, specialty and years of experience must be recorded.
- Each pharmacy has a name, address and phone number. A pharmacy must have a manager.
- A pharmacist is identified by an SSN, he/she can only work for one pharmacy. For each pharmacist, the name, qualification must be recorded.
- For each drug, the trade name and formula must be recorded.
- Every patient has a primary physician. Every doctor has at least one patient.
- Each pharmacy sells several drugs, and has a price for each. A drug could be sold at several pharmacies, and the price could vary between pharmacies.
- Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors. Each prescription has a date and quantity associated with it.

[Solution]

Produced: 26 Jul 2016