COMP3311 (07s2) Course Introduction

Units of credit	6
Parallel teaching	no

1. Pre-requisites

The formal pre-requisite for this course is COMP2011 Data Organisation. The knowledge that we assume from COMP2011 is: experience with procedural programming, and an understanding of a range of in-memory searching data structures (e.g. binary search trees, 2-4 trees, and hash-tables).

Some of the other material in the course (e.g. ER model and basic SQL) may also be familiar to students who have taken these courses, but don't make the mistake of thinking “I know all this stuff”; we will most likely cover these areas in more depth than you've seen them previously.

2. Course philosophy and teaching strategies

The learning foci in this course are primarily lectures and tutorials (theoretical knowledge) and laboratories and projects (practical knowledge).

The course will have an emphasis on problem solving for real applications.

Students will learn the main contents of the course through lectures. Tutorials are available to assist students to obtain in-depth understanding of course materials and develop problem solving skills by working on tutorial questions.

3. Course aims

This course aims to explore the practice of developing database applications, the theory behind relational database systems, and the technologies used to implement database management systems. This course will be the foundation of other database courses, such as COMP9315 (DBMS Implementation), COMP9314 (Next Generation Database Systems), COMP9318 (Data Warehousing and Data Mining), COMP9321 (E-Commerce Systems Implementation Infrastructure), etc.

4. Syllabus

The following is a summary of the topics that will be covered in this course. Topics will not be covered in the order shown here. There will be different levels of detail on the different topics (some will provide overviews, some will be more detailed).

1. Data modelling and database design
    - Entity-relationship (ER), relational models
    - Relational database design (theory of dependencies, normalisation, ...)
2. Database application development
    - SQL for querying and data definition (PostgreSQL's version)
    - SQL for constraints and triggers (PostgreSQL's version)
3. DBMS implementation technology
    - DBMS architecture
    - Storage and indexing, query evaluation, transaction processing
    - Database performance tuning

5. Learning outcomes

Students successfully completing this course will be able to:

- develop accurate, non-redundant data models
- realise data models as relational database schemas
- implement relational schemas via the Oracle DBMS

- formulate queries via the full range of SQL constructs
- effectively utilise triggers and stored procedures
- understand performance issues in relational database applications
- understand query and transaction processing techniques for RDBMSs
- appreciate the past, present and future of database technology

The learning outcomes are closely related to UNSW graduate attributes 1 -- 6. For example, bonus questions will be given in the assignment in order to encourage independent learning and critical thinking.

6. Administrative Components

See the course hompage (http://www.cse.unsw.edu.au/~cs3311) for up-to-date information regarding **Course Staff**, **Course Schedule**, and **Course Resource List**.

As the course is continuously developing together with the data warehousing and data mining fields, the course schedule is subject to change too. Please read the introduction slides (lecture notes of the first week) for the course schedule in the current offering.

7. Assessment

```
quiz1  = mark for quiz 1
quiz2  = mark for quiz 2
quiz3  = mark for quiz 3
quiz4  = mark for quiz 4
quiz5  = mark for quiz 5
quiz6  = mark for quiz 6
ass1   = mark for written assignment 1
proj1  = mark for programming project 1
proj2  = mark for programming project 2
exam   = mark for final exam

Q      = (quiz1 + quiz2 + quiz3 + quiz4 + quiz5 + quiz6) / 6
t      = (Q + ass1 + proj1 + proj2) / 4
grade  = (2 * exam * t) / (exam + t)
```

Both written and programming assignments are helpful to achieve deep understanding of the course materials and develop problem solving abilities.

Relationship to the Learning Outcomes:

- All assignments and the final examination are used to assess whether you have achieved the objectives of the subject.
- The programming assignment, in particular, gives you the opportunities to engage in real problems, and helps you to develop in-depth understanding of the course materials and develop problem-solving skills.

Grading Criteria: Grading criteria for each assessment will be detailed in the specification.

Late submission: Quizzes/assignments/projects submitted late are subject to late penalties, which are specified in the assignment/project specifications. “Soft” Late penalties are used in this course, which only reduces the maiximum mark obtainable. Thus if the assignment is marked out of 10, and students A and B hand in assignments worth 9 and 7, both receiving 20% penalty, then the maximum mark obtainable is 8, so A gets min(9, 8) = 8 and B gets min(7,8) = 7.

Assignment submission: Assignment submission procedure is described in the assignment specification document, which will be linked to this page when the assignment specification becomes available. Generally assignments are submitted electronically using the give program running on the School's computer systems (in labs, and on servers). Details are in the assignment specifications.

8. Reading e-mail

You should check your school e-mail frequently in case of announcements relating to this course. We assume that you read e-mail sent to your CSE account by the next working day during teaching sessions.

- How to forward your CSE E-mail to another e-mail address
- CSE E-mail guidelines

9. Academic honesty and plagiarism

Copying assignments is unacceptable. Assignments will be checked. The penalties for copying range from receiving no marks for the assignment, through receiving a mark of 00 FL for the course, to expulsion from UNSW (for repeat offenders). Allowing someone to copy your work counts as plagiarism, even if you can prove that it is your work.

Further details of the School plagiarism policy can be found here. (You acknowledged receipt of these rules when you obtained your CSE computer account, and the link above is for your convenience so that you can review the rules now.)

We are aware that a lot of learning takes place in student conversations, and don't wish to discourage those. However, it is important, for both those helping others and those being helped, not to provide/accept any programming language code in writing, as this is apt to be used exactly as is, and lead to plagiarism penalties for both the supplier and the copier of the codes. Write something on a piece of paper, by all means, but tear it up/take it away when the discussion is over.

If you are new to studying in Australia, be aware that attitudes to plagiarism at UNSW may be different from those in your home country. Make sure you are clear about the rules here at UNSW. In brief, and for the purposes of COMP9444, plagiarism includes copying or obtaining all, or a substantial part, of the material for your assignment, whether programming language code, or written or graphical report material, without written acknowledgement in your assignment from:

- location on the internet;
- a book, article or other written document (whether published or unpublished) whether electronic or on paper or other medium;
- another student, whether in your class or another class;
- a non-student (e.g. from someone who writes assignments for money)

Note that if you copy code or other material from another student or non-student with acknowledgement, you will not be penalised for plagiarism, but you are unlikely to get any marks for the copied material. If you use code found in a publication (on the internet or otherwise) then the marks you get for this will be at the marker's discretion, and will reflect the marker's perception of the amount of work you put into finding and/or adapting the code, and the degree to which you understand the code.

Note also that there is a big difference between being able to understand someone else's code, and writing that code yourself from scratch. A computer programmer has to be able to write code from scratch. The assignments provide opportunities for you to develop the skills necessary to write your own neural-network-related code. Use these opportunities!

10. Course Material

- All lecture notes, tutorial and lab notes will be posted on the course web site. These notes summarises the major contents and help you to understand the materials when you read the textbook later. You definitely need to read the corresponding chapters in the textbook to gain a full understanding of the course materials.
- [SKS05] is the textbook for this course. We got two copies on **standard loan** in the library, two on **one week loan**, and four on **two hour loan**.

```
[SKS05] Database System Concepts, Silberschatz, Korth, Sudarshan
        5th edition, 2005, McGraw-Hill.
```

- Reference books for this course are:

```
[RG02]  Database Management Systems, Ramakrishnan and Gehrke
        3rd edition, 2002, McGraw-Hill.
```

- Programming with PostgreSQL: Several links to the oracle documents and manuals will be provided in the course homepage. We will not be able to cover them in detail and you are expected to learn by yourself. Searching the web (esp. googling) is also a great way to find answers for beginners.

11. Further information

- Students enrolled in COMP3311 are expected to attend all classes
- The use of School of Computer Science and Engineering computing laboratories is subject to rules described in the Yellow Form, which you acknowledge (electronic) receipt of when you receive your computing account. The Yellow Form also outlines what to do in case illness or misadventure that affects your assessment, and supplementary examinations procedures within the School of Computer Science and Engineering. -->
- Information on UNSW Occupational Health and Safety policies and expectations
- Equity and Diversity issues

12. Continual Course Improvement

Each year feedback is sought from students and other stakeholders about the courses offered in the School and continual improvements are made based on this feedback. UNSW's Course and Teaching Evaluation and Improvement (CATEI) Process (http://www.ltu.unsw.edu.au/ref4-5-1_catei_process.cfm) is one of the ways in which student evaluative feedback is gathered. Significant changes to courses and programs within the School are communicated to subsequent cohorts of students.

Wei Wang, June 2007