COSC 404
Database System Implementation

Course Introduction

Dr. Ramon Lawrence
University of British Columbia Okanagan
ramon.lawrence@ubc.ca

The Essence of the Course

If you walk out of this course with nothing else you should:

Understand database algorithms and techniques in order to:
1) Be a better, "expert" user of database systems.
2) Be able to use and compare different database systems.
3) Adapt the techniques when developing your own software.

This course opens the database system "**black box**".

My Course Goals

My goals in teaching this course:

- Summarize and present the information in a simple, concise, and effective way for learning.
- Strive for ***all*** students to understand the material and pass the course.
- Be available for questions during class time, office hours, and at other times as needed.
- Provide a background on the fundamental concepts of database systems including transactions and concurrency.
- Create opportunities to learn concepts by experimenting and programming with different database systems.
- Encourage students to continue studying databases including further projects and graduate level research!

Course Objectives

1) To learn how to manipulate data in memory and secondary storage and use index structures for improved performance

2) To understand the steps of query processing including parsing, translation, optimization, and execution

3) To understand the principles of transactions, concurrency, recovery, and distribution as they apply to databases

4) To apply fundamental knowledge of database techniques to be better users with the ability to use different database systems and compare their properties

Your Course Goals

Your goals in taking this course:

- To sufficiently learn the material to pass the course.
- To learn algorithms and techniques that constitute the ***foundations*** of database theory and implementation.
- To understand how a database system works in order to better understand how to use them properly.
- To realize that database technology is present in many areas including operating systems, networks, and programming.
- To form a background knowledge on databases, and determine if you want to continue with database related research.
- To develop experience in using a variety of database systems.

Academic Dishonesty

Cheating in all its forms is strictly prohibited and will be taken very seriously by the instructor.

A guideline to what constitutes cheating:

- Assignments
  - ⇨ Working in groups to solve questions and/or comparing answers to questions once they have been solved.
  - ⇨ Discussing HOW to solve a particular question instead of WHAT the question involves relative to the notes.
  - ⇨ Copying code, even small code fragments, from other students.
  - ⇨ You may discuss general ideas and syntax, but never share code!
- Exams
  - ⇨ All exams are closed book, so no course materials should be present.

Academic dishonesty may result in a "F" for the assignment or course and ***all*** instances are recorded in the Dean's office.

Assignments

There will be weekly written and programming assignments.

Written Assignments (15% of overall grade):

- Practice questions similar to midterm/final exams.
- Will have some time in class but mostly as homework.

Programming Assignments (20% of overall grade):

- Experience applying concepts to a variety of database systems.
- Will be mostly done in lab but may take more than 2 hours.

Both written and programming assignments can be done individually or in pairs.

The assignments are critical to learning the material and are designed to prepare you for the exams!

The In-Class Clicker Questions

To encourage attendance and effort, ***5%*** of your overall grade is allocated to answering in-class questions using a clicker.

- The clicker can be purchased at the bookstore and sold back to the bookstore like a used textbook.
- The clicker is personalized to you with your student number.
- At different times during the lectures, questions reviewing material will be asked. Reponses are given using the clickers.

There will be at least 60 questions throughout the semester. Each question is worth 1 mark, and you need at least 50 right answers to get the full 5%.

- That is, if you answer 40 questions right, you get 40/50 or 80%.
- No make-ups for forgetting clicker or missing class.

Database Implementation Project

For graduate students only:

20% of your mark is for a major database development project.

Goal of the project is to experiment with new database systems or experiment with novel techniques expanding on class material.

This is ***not*** implementing a web site with a relational database like COSC 304.

How to Pass This Course

The most important things to do to pass this course:

- Attend class
  - ⇨ Read notes *before* class as preparation.
- Do the written assignments
  - ⇨ Important practice to learn the material for the exams!
- Spend time doing the programming assignments
  - ⇨ Programming with databases is a valuable, employable skill.

To get an "A" in this course do all the above plus:

- Do additional practice questions.
  - ⇨ Practice questions are especially helpful to re-enforce concepts.
- Spend additional time programming
  - ⇨ Programming assignments may take longer than a lab time. Extra time invested will payoff significantly in grades and future jobs.

Systems and Tools

Connect is used for a discussion board, for posting marks, and for anonymous feedback.

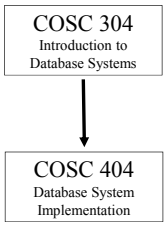
- Please use the discussion board and feedback survey.

All software is available in the laboratory at SCI 234.

Access to database systems will be provided as needed.

These systems will have separate user ids and passwords.

COSC 304 vs. COSC 404



Database Design and Programming

- Data models - ER, relational, XML, JSON
- Query languages - SQL, relational algebra
- Design project
- Database skills and techniques as a user
- How to use a DBMS ; how to build a database

Database System Implementation

- Storage and index structures
- Transaction management, concurrency control
- Query processing, recovery and reliability
- How to build a DBMS
- Non-relational systems and architectures
- How to select a DBMS

Why are you here?
Reasons Why People Take This Course

A) I need an upper-year Computer Science elective, and this course was all there was…

B) I liked COSC 304 (Intro. Databases) and thought this course may be okay too.

C) I am curious about what is in the database "black box".

D) I want to be a better developer and database user to improve my skills for future jobs.

E) I am interested in database research and advanced studies.

What to Learn
What Topic are You Most Interested In?

A) Accessing data on hard drives and solid state drives

B) Learning how SQL queries get processed inside a database system

C) Learning how a database handles multiple users and recovers from failures

D) Experimenting with different databases like PostgreSQL, MongoDB, and MySQL

E) None of the above

What do you expect?
What Grade are You Expecting to Get?

A) A

B) B

C) C

D) D

E) F

My Expectations

My goal is for you to learn the material and walk out of this course confident in your abilities:

- To understand how a DBMS is constructed
- To make intelligent decisions on data allocation, indexing, and physical designs
- To describe how a DBMS supports concurrent users, transactions, and recovers from failure

I have high standards on the amount and difficulty of material that we cover. I expect a strong, continual effort in keeping up with readings and doing assignments.

The course will be very straightforward – If you do the work, you will do well.

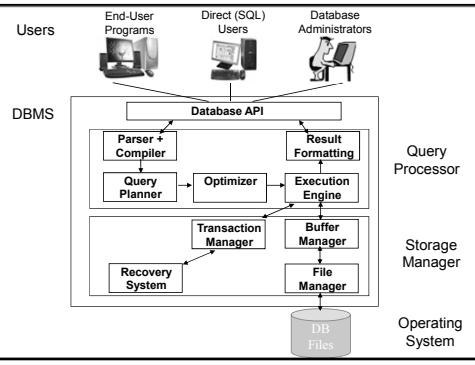
Your mark is 60% perspiration and 40% inspiration.

Database System Implementation
Motivation

Key requirements of a database system:

- 1) Data Storage and Persistence:
  - ⇨ How is data organized? Where is it located?
- 2) Query Processing:
  - ⇨ How does the user query the data? How efficient is it?
- 3) Transactions, Consistency, and Reliability:
  - ⇨ What happens if the computer crashes while the user is updating data?
- 4) Concurrency:
  - ⇨ Can multiple users access the data at the same time? What happens if multiple users update the same data item?
- 5) Security:
  - ⇨ How do you verify the user has access to the data?
- 6) Scalability:
  - ⇨ How do you handle Big Data and lots of users?

Traditional
Database System Architecture



Databases Architectures Not "One Size Fits All"

Relational databases (RDBMS) are still the dominant database architecture and apply to many data management problems.

- Over $20 billion annual market in 2015.

However, recent research and commercial systems have demonstrated that "one size fits all" is not true. There are better architectures for classes of data management problems:

- Transactional systems: In-memory architectures
- Data warehousing: Column stores, parallel query processing
- Big Data: Massive scale-out with fault tolerance
- "NoSQL": simplified query languages/structures for high performance, consistency relaxation

COSC 304 Review Question

Question: What was the acronym used to describe transactional processing systems?

A) TP

B) OLAP

C) OLTP

D) DBMS

Research Question

Question: What company is the largest database software vendor by ***sales volume***?

A) Microsoft

B) Oracle

C) IBM

D) Google

Database Architectures: NoSQL vs Relational

"NoSQL" databases are useful for several problems not well-suited for relational databases with some typical features:

- **Variable data:** semi-structured, evolving, or has no schema
- **Massive data:** terabytes or petabytes of data from new applications (web analysis, sensors, social graphs)
- **Parallelism:** large data requires architectures to handle massive parallelism, scalability, and reliability
- **Simpler queries:** may not need full SQL expressiveness
- **Relaxed consistency:** more tolerant of errors, delays, or inconsistent results ("eventual consistency")
- **Easier/cheaper:** less initial cost to get started

NoSQL is not really about SQL but instead developing data management architectures designed for scale.

- NoSQL – "Not Only SQL"

Example NoSQL Systems

MapReduce – useful for large scale, fault-tolerant analysis

- Hadoop, Pig, Hive

Key-value stores – ideal for retrieving specific items from a large set of data (architecture like a distributed hash table)

- high-scalability, availability, and performance but weaker consistency and simpler query interfaces
- Cassandra, Amazon Dynamo, Google BigTable, HBase

Document stores – similar to key-value stores except value is a document in some form (e.g. JSON)

- MongoDB, CouchDB

Graph databases – represent data as graphs

- Neo4J

Survey Question

Question: Have you used any database system besides MySQL and Microsoft SQL Server used in COSC 304?

A) Oracle

B) MongoDB

C) PostgreSQL

D) More than two different databases used

E) No other databases used

Why this Course is Important

DBMS technology has applications to any system that must store data persistently and has multiple users.

- Even if you will not be building your own DBMS, some of your programs may need to perform similar functions.
- The core theories expand on topics covered in operating systems related to concurrency and transactions.

A DBMS is one of the most sophisticated software systems.

- Understanding how it works internally helps you be a better user of the system.
- Understanding of database internals is valuable if you will perform database administration duties or be responsible for deciding on a database architecture for an application.

Database technology is a key component of our IT infrastructure that will continue to require innovation in the future.