ECE 656 Database Systems (Fall 2021)

Paul A.S. Ward

August 4, 2021

This is an introductory graduate course on database systems. It covers what a database system is, why it is useful, and how it is used. In particular, this course will describe in detail the relational data model and how it enables effective problem solving, the logical and physical layers of database systems in general and of relational database systems specifically, and correct data modeling and database design. It also covers transactions, data persistence, and system dependability, both in terms of proper operation as well as system implementation and the implications of that realization. Data mining and system administration are also covered. Time permitting, the course will give a brief introduction to advanced topics in databases.

Students will apply their learning through a series of assignments and a sizable summative project that will demonstrate their ability to design relational schema using a modern RDBMS, answer queries, resolve performance problems, and other aspects of database systems. All aspects of database design, from data modeling through to efficient implementation will be required, as well as data analytics and database administration.

1 Learning Outcomes:

By the end of this course, students who have satisfactorily completed the course requirements should be able to:

1. Understand Entity-Relationship Models (ERM) (entities, constraints, relationships, cardinality, primary keys)
2. Create an appropriate ERM given a broad set of requirements
3. Understand the Relational Model (RM), Relational Algebra (RA) and extended relational algebra, and 3-valued logic, and be able to solve problems using extended relational algebra
4. Transform an ERM to a RM
5. Recognize and create good relational designs for data, including functional dependency, multi-functional dependency, normal forms, and relational decomposition
6. Recognize poor relational designs and explain why they are poor
7. Understand the basics of SQL, especially DML and DDL, but also DCL and TCL, and its integration into client-application code
8. Transform a RM to SQL and RA to SQL
9. Understand and resolve performance problems in databases
10. Understand the basics of transactions and the relationship between transactions and data integrity
11. Understand the basics of modern database architecture, includ-

ing indexing, query optimization, data storage, concurrency control, transaction management, and failure recovery
12. Understand the significance of database architecture for good relational design and apply that to ensure good relational design
13. Understand the basics of database administration, setting up and managing a database, and security basics
14. Understand the basics of data mining and be able to implement one of the three main data-mining algorithms

2 *Prerequisites:*

Set Theory: especially relations, functions, closure, and proofs
Logic: especially propositional logic, operators, and predicate logic
Data Structures: especially B-trees and abstract data types
Algorithms: especially sorting, hash functions, and Big-O notation
Systems Programming: especially memory and file management, but also processes, threads, inter-process communication, *etc*.
Language Arts/English: syntax, semantics, grammar, composition
Practice: recognizing that practice is essential to learning

It is implicitly assumed because of the above that students have knowledge of the C programming language and some variant of un*x. Knowledge of a scripting language is extremely helpful (*e.g.*, bash, Perl, *etc*.). It is likewise assumed that students have access to suitable computer equipment, though database-server machines will be accessible for students taking this course.

3 *Course Website*

All information will be posted on Learn and/or Piazza. Learn will be used for the two tests we will have through the term. We will use Piazza for surveys and as a discussion forum. The Piazza course website is:

`https://piazza.com/class/krw9yyf4icf5vo`

The average response time for questions asked on Piazza is typically less than half-an-hour, depending on the time of day, day-of-the-week, and the willingness of your fellow students to engage in the discussion. Do not expect a rapid response to any other form of communication (e-mail, Learn posting, *etc*.). You will be enrolled into this Piazza course in early September and will receive an e-mail requesting you sign up to Piazza. If you do not receive such an e-mail by September 8th, please notify the course staff so this can be resolved.

Ask questions on Piazza. Questions asked elsewhere will be redirected to Piazza.

4 *Textbook*

Reference Text: Silberschatz, Korth, and Sudarshan, *Database System Concepts*. McGraw-Hill. 7th edition. 2019.
Reference: C.J. Date and Hugh Darwen, *Databases, Types, and The Relational Model: The Third Manifesto*. Available at `https://www.dcs.warwick.ac.uk/~hugh/TTM/documents_and_books.html`
Additional reference material will be added as appropriate to the course requirements.

The 6th edition is also adequate for most of the course material and in some instances is superior.

Slides will be posted on Learn and (at the instructor's discretion) on Piazza. However, the slides *WILL NOT*, in general, contain sufficient information to understand the course material. You will need to supplement it by some mixture of class attendance, problem solving, and reading the text and/or reference material.

5 *Lecture Schedule*

In the following table are the lecture topics by week and the objective(s) to which they contribute.

Week	Topic	Objective(s)
1, 2	Purpose; relational model and algebra; eRA queries	3
3, 4	Logical layer; SQL DML and DDL language basics; transforming eRA to SQL	7, 8
5, 6	Database design: ER modeling; transforming ERM to RM; atomicity; functional dependencies, multivalued dependencies, and relational decomposition	1, 2, 4, 5, 6
7, 8	Physical layer: storage, buffer-pool, file organization, indexes, query evaluation	9, 11
9, 10	Transactions, data persistence, and system dependability	10, 11, 12
11	Data mining: supervised and unsupervised analysis techniques; classification, clustering, and correlation	14
12	Advanced topics (text databases, key-value stores, warehousing, ...)	

Administration and security issues (12) will be covered throughout the term rather than as a distinct section, as it is necessary for students to learn the basics of administration from very early in the course.

Given that this course is being run online, lecture content will be in the form of video lectures on Learn. The scheduled meeting times [redacted]) will be used for synchronous verbal Q&A, starting in the first week of term; Piazza provides for asynchronous text-based Q&A. Lecture videos will be posted to Learn in approximately the above lecture order, but will likely be somewhat ahead of schedule so that you may find it useful to view the material, and read the corresponding textbook sections, in a somewhat different sequence than the above.

6 Course Deliverables

The course deliverables comprise three assignments (25%), two in-class tests (25%), and a two-person project (50%).

Assignments

The course has three assignments that collectively work towards, but are distinct from, the course project, as well as preparing students for the tests. These assignments will be a mixture of problem solving with extended relational algebra, SQL coding, database design, and performance analysis. The assignment component of the course is worth 25%, with valuations as described below. Assignments are to be completed individually and will be subject to plagiarism analysis.

The assignment schedule is as follows:

No	Value	Handout	Due	Objectives
1	8%	Sept 15th	Oct 7th	Admin, RA, SQL
2	7%	Oct 6th	Oct 28th	DB Design and Evolution
3	10%	Oct 27th	Nov 25th	Performance and Analytics

Assignments are due at noon, Waterloo time, on their due date. The clock on the submission server (Learn, marmoset, elsewhere) will be deemed to be correct for the purpose of determining whether or not an assignment was received on time.

The assignment submission server and/or various data networks may be busy. Do not leave your submission to the last minute.

Tests

There are two tests in the course, scheduled as follows:

No	Date	Materials Covered
1	Nov 4th	Relational algebra, logical schema, design
2	Dec 2nd	Physical schema, dependability, analytics

The tests are worth 25% of your final grade, weighted equally. The tests will be delivered online through Learn.

Project

The course has a project worth 50% of your final grade. The project is to be completed in groups of two. The detailed requirements for the project are provided in separate documents, but broadly speaking you are required to propose a non-trivial database-systems application that uses a sizable dataset of real-world data, will require client-side coding, server-side coding, and data analytics. Examples of possible projects include:

Given that the number of students enrolled in the class may not be an even number, there may also be one group of three students or of a single student. The evaluation of the project for that group will take this into account.

Better Aggregation: Add better aggregation functionality to the MySQL database by creating user-defined functions that could compute median, provide confidence intervals (especially in the presence of NULLs), *etc*. Such a project would need to get a sizable dataset and demonstrate its abilities on that dataset.

CLI for Postgres: The Postgres database has `psql` for a command-line interface. However, this is a much less friendly interface than the MySQL CLI. Postgres also lacks a LOAD utility. Such a project would create a MySQL-style CLI and LOAD utility for Postgres.

Car-Sale Application: Several gigabytes of car-sale records are available on Kaggle. Designing and implementing an autotrader-like application can be done and then populated with such data; a data-analysis exercise can then be performed on the dataset.

More details are provided in the project-description documents, but this is provided here to offer a few examples for students wanting some idea of the kinds of projects that would be expected.

The project deliverables have the following schedule:

Item	**Due Date**	**Expected Outcome**
Team Formation	Sept 15th	Find a partner for the project
Initial Proposal	Sept 22nd	Half-page project proposal with sources
Proposal Feedback	Sept 29th	Accept/Modify/Start Again
Final Proposal	Oct 6th	Final proposal must be approved
Final Deliverables	Dec 23rd	Final project deliverables due

The final deliverables are due on the last day of exams, at 4:30 PM, Waterloo time. This schedule means that you should have a minimum of 11 weeks in which to complete the project. Only the final deliverables are graded. These final deliverables are as follows:

1. Written Report: in the format of a conference/journal submission
2. Code: in university git repository; must include testcases
3. Video Demo: a half-hour walk-through/presentation of your project; it should be done in the style of a conference presentation

While there are no intermediate deliverables for the project, it is highly recommended that you obtain informal feedback during the course of working on your project (particularly with respect to design issues) so as to ensure that you are both on track and on the right track.

7 *Course Staff*

Instructor: Paul A.S. Ward
TAs: ?

Contact information is deliberately omitted. The preferred contact techniques are:

- Ask a question during the synchronous class meeting time
- Ask a question on Piazza

If you do not want others to see your question, you can ask the instructors/TAs specifically and/or make the question anonymous.

8 Grading Scheme Summary

There are three evaluation components in the course with the following weight:

- Project: 50%
- Assignments: 25%
- Tests: 25%

9 Course Policies

Late Submissions

Late assignment and project submissions are not accepted. The timestamp on Learn and or the auto-grader will be considered the official time of submission of any lab. Do not leave things to the last minute.

Published research has shown that exam performance in is inversely correlated with starting time of assignment submission: the later you start, the lower your exam grade is likely to be.

Missing Something

If you miss the midterm exam and are able to provide a good reason, the grading scheme will change such that the final exam is (nominally) worth 60%. If you miss the midterm exam without a good reason, you will get 0 on the midterm and the grading scheme above still applies. The University rules say if you miss the final exam, without an acceptable reason, your grade in the class will be "DNW - Did Not Write." This is very undesirable. Show up for the final exam.

Good reasons include: illness (submit a Verification of Illness (VoI) form), coop job interview (submit evidence of interview), *etc.*

If you fail to submit your homework, assignment submission, *etc.*, by the required deadline, any valid excuse must cover the entire period from the distribution of the item to the class up till the submission time for the total weight of the missing item to be shifted to the other homework assignments. If you have a valid excuse just for the day the homework is due, only an unscheduled justification (*e.g.*, illness) will be valid, and it will only be used to reduce a portion of the weight of the missing homework. Any scheduled activity on the submission day is presumed to be something that you should plan around, and will not be considered a valid excuse for failure to submit your homework.

Collaboration & Plagiarism

Plagiarism, taking credit for work that others did, is not permitted, and this applies to source code as well as tests/exams. The course staff will be checking for it through a variety of different methods. Any cases of plagiarism detected will be reported, according to university policy (see the University Policy section below).

You may discuss ideas, algorithms, problems, possible solutions, *etc.*, and help other students debug small fragments of code. However, each student must submit his/her own, independently developed code for each assignment and project. While it can be very useful to look at someone else's code you should not be doing that anywhere you might be writing your own code.

Students are not permitted to share code, whether electronically or in written form.

The University of Waterloo takes the issue of plagiarism very seriously (See UW Policy 71). If you are uncertain about this subject, please seek some guidance. There are many resources available to you. You can check the university policies, talk to your course instructor or support tutor, etc.

To sum this up in two short instructions:

1. Acknowledge the work of others.
2. If you are uncertain, ask!

In the case of the group-project work, your group will submit its own independently developed code

You cannot become a great writer without reading the works of other great writers; likewise for coding.

G-d is watching and so are we: all code is subject to plagiarism analysis.

Late Submissions

Late assignments and project submissions will not be accepted.

Re-marking

If you believe that your grade on an a written, submitted deliverable (*e.g.*, a midterm exam question) is incorrect or unfair, you may ask that it be re-marked. Any request for remarking must be made within two weeks of the deliverable being returned to you. While you are free to identify the portion of the deliverable that you believe to be incorrectly graded, you should be aware that any regrading will be done on the entire submission. The reason for this is that while we acknowledge that mistakes can occur in which students are unfairly penalized, it is also the case that mistakes are made in which students receive higher grades for a question than is warranted. As such, any regrading may result in your grade going up, down, or remaining the same.

Please be aware that midterms are scanned before being handed back to students. As such if you attempt to add or change an answer your wrote on a midterm after it is handed back to you and then

We're not going to correct a mistake that cost you 3 marks on one question and ignore the fact that a grading error in your favour gave you 5 undeserved marked on a different question.

Yeah, about that: we don't actually tend to lower grades when you request a regrade, even if it is justified. I have to write this here, though, to reserve the right to do so in rare and exceptional circumstances.

request a regrade, this is considered by the university to be cheating on an exam. This is very, very bad. Do not do this.

If the error you identify is simply one of an incorrect addition of the marks, then the correction will be made without requiring regrading.

Attendance & Illness

It is usually a good idea to attend lectures. In particular, the average of the grades of students who attend lectures/labs/tutorials is about 20% higher than that of those who do not attend. That said, this is university and you are capable of deciding for yourself if you are going to the lecture or not. Attendance is not taken and not graded.

Some submissions may be required during labs, as identified above. In such instances you are required to attend.

During the term, you may need arrive late to a class or leave partway through, because of co-op interviews. This is not a problem, as long as you are not disruptive when arriving/departing. Be mindful of your classmates.

If you feel ill, you should seek appropriate medical attention. If you miss an exam for health reasons, you need a Verification-of-Illness (VoI) form. Forms can be completed by the physicians at Health Services. Your completed verification of illness form should be presented to the First-Year Engineering Office and not directly to the course staff. If you anticipate missing a deliverable deadline or an examination for a non-medical reason, you should contact me as soon as you are aware of the problem. Given sufficient notice, alternate arrangements may be possible.

10 University Policies

Academic Integrity

In order to maintain a culture of academic integrity, members of the University of Waterloo community are expected to promote honesty, trust, fairness, respect and responsibility. For more information check:
`https://uwaterloo.ca/academicintegrity/`

Grievance

A student who believes that a decision affecting some aspect of his/her university life has been unfair or unreasonable may have grounds for initiating a grievance. Read Policy 70, Student Petitions and Grievances, Section 4:
`uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-70`
If in doubt, contact the department's administrative assistant, who will provide further assistance.

Discipline

A student is expected to know what constitutes academic integrity (see above section) to avoid committing an academic offense, and to take responsibility for his/her actions. A student who is unsure whether an action constitutes an offense, or who needs help in learning how to avoid offenses (*e.g.,* plagiarism, cheating) or about "rules" for group work/collaboration should seek guidance from the course instructor, academic advisor, or the undergraduate Associate Dean. For information on categories of offenses and types of penalties, students should refer to Policy 71 (Student Discipline):
`adm.uwaterloo.ca/infosec/Policies/policy71.htm`
For typical penalties, check the Policy 71 appendix "Guidelines for the Assessment of Penalties":
`www.adm.uwaterloo.ca/infosec/guidelines/penaltyguidelines.htm`

Appeals

A decision made or penalty imposed under Policy 70 (Student Petitions and Grievances) (other than a petition) or Policy 71 (Student Discipline) may be appealed. A student who believes s/he has grounds for an appeal should refer to Policy 72 (Student Appeals):
`uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-72`

Privacy

Questions about the collection, use, and disclosure of personal information by the University, should be directed to the Freedom of Information and Privacy Coordinator, Secretariat, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1. The email address of the Freedom of Information and Privacy Coordinator is fippa@uwaterloo.ca. See also University of Waterloo Policy 46: Information Management
`uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-46-information-management`

Note for Students with Special Needs

AccessAbility Services (AAS) located in Needles Hall, Room 1403, collaborates with all academic departments to arrange appropriate accommodations for students with disabilities without compromising the academic integrity of the curriculum. If you require academic accommodations to lessen the impact of your disability, please register with the AccessAbility Services office at the beginning of each academic term.