CMSI 486-01, -02

Introduction to Database Systems

https://dondi.lmu.build/fall2020/cmsi486

Fall 2020—online; see Brightspace for links
MW 4–5:15pm (01), 2–3:15pm (02), 3 semester hours
Office Hours: MW 1–2pm, 5:15–6pm; T 2–3:30pm, 5–6pm;
or by appointment *(don't hesitate to ask!)*

John David N. Dionisio, PhD
email: *dondi@lmu.edu*
Virtual Doolan 102; (310) 338-5782

Objectives and Outcomes

This course introduces the computer science subfield of *databases*, which is concerned with the theory, design, and implementation of systems that manage large amounts of data. Long after the course concludes, my hope is that you will:

1. **Know and understand how databases are designed, implemented, and deployed.**
2. **Be acquainted with database system theory and algorithms.**
3. **Apply this knowledge by designing and implementing a database application library for a particular domain.**

In addition to the course-specific content, you are also expected to:

4. **Follow disciplinary best practices throughout the course.**

Prerequisites/Prior Background

Although there are no absolute prerequisites to this course, students will benefit greatly from having taken CMSI 386 Programming Languages and CMSI 387 Operating Systems. Intermediate to advanced programming proficiency in any language will be helpful, as well as familiarity and experience with the command line interaction style.

Materials and Texts

This course does not have a preassigned textbook, with materials consisting primarily of assorted websites, articles, videos, and sample code to be made available online. However, the following can serve as foundational reading:

- Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*, 7th Edition, Pearson, 2016.

Having the latest exact edition isn't critical, so no problems if an older edition is more cost-effective for you.

Course Work and Grading

Your final grade will be based on the percentage of the points you get for the following deliverables against the total number of possible points:

Database fiddle tutorial	100
File database mini-stack	100
Relational database mini-stack	100
Document database mini-stack	100
Graph database mini-stack	100
Full database SDK	200
Total	700 points

Percentages ≥ 90% get an A– or better; ≥ 80% get a B– or better; ≥ 70% get a C– or better. I may nudge grades upward based on qualitative considerations such as degree of difficulty, effort, class participation, time constraints, and overall attitude toward the course.

Term Portfolio

Your accumulated assignments for the semester comprise the *term portfolio*—the final, definitive artifact that demonstrates the course's outcomes. It is how you show whether you have, indeed, accomplished the objectives of this course.

An assignment's number is its due date in *mmdd* format, and it is always due by 11:59:59.999pm of that date. Point values are based on the state of your assignments at that moment.

Your portfolio for this course will consist of the following types of deliverables:

- Database fiddle tutorial—a hands-on demonstration of basic database operations on an online "fiddle" system (*1a–1c*, *3c*)
- "Mini-stacks"—case studies where data sets are loaded and managed using a range of database paradigms: modeling, CRUD, and the beginnings of a persistence layer API (*1a–1d, 3a–3d*)

- Full database SDK—a full-scale persistence layer implementation consisting of a data model, CRUD implementations, and a small demonstration application in a chosen data set's domain (*2b, 3a–3d*)

Outcomes *4d–4f* apply to all assignments; the full range of *4a–4f* applies to all assignments that involve programming.

For maximum enthusiasm and interest level, you will be allowed to select a particular application domain (assuming there is a robust data set available for it) around which your assignments can revolve. The hope is that such a selection will allow you to formulate database queries and implement database operations that you will find to be personally interesting and fulfilling. Students who chose the same application domain may form a "guild" of sorts where they can gain collective expertise in that domain and its data.

Version Control

Version control is an indispensable part of today's computer science landscape in industry, the academe, and the open source community. We use version control heavily in this course: make sure that you get the hang of it.

None of the assignments can be completed (well) overnight; they should be the result of steady progress from the moment they are assigned to the date they are due. "One and done" submissions will negatively affect the final score.

Workload Expectations

In line with LMU's *Credit Hour Policy*, the workload expectation for this course is that for every one (1) hour of classroom instruction (50 scheduled minutes), you will complete at least two (2) hours of out-of-class work each week. This is a 3-unit course with 3 hours of instruction per week, so you are expected to complete $3 \times 2 = 6$ hours of weekly work outside of class.

Attendance

Attendance at all synchronous sessions is ideal, but not required. If you must miss class, it is your responsibility to notify me about this and keep up with the course asynchronously.

Due to the remote/online format of the class, it can be tempting to "spoof" yourself by logging in, deactivating audio and video, and ultimately tuning out. If you're planning to do this, don't bother—just watch the session recording later. Better for the class to know that you really aren't around than for us to *think* you're around, only to realize that you aren't when we try to interact with you.

The last day to add or drop a class without a grade of W is September 4. The withdrawal or credit/no-credit deadline is November 6.

Academic Honesty

Academic dishonesty will be treated as an extremely serious matter, with serious consequences that can range from receiving no credit to expulsion. It is *never* permissible to turn in work that has been copied from another student or copied from a source (including the Internet) without properly acknowledging/citing the source. It is your responsibility to make sure that your work meets the standard of academic honesty set forth in:

http://academics.lmu.edu/honesty

Americans with Disabilities Act

Students with special needs who require reasonable modifications, special assistance, or accommodations in this course should promptly direct their request to the Disability Support Services (DSS) Office. Any student who currently has a documented disability (ADHD, Autism Spectrum Disorder, Learning, Physical, or Psychiatric) needing academic accommodations should contact the DSS Office (Daum Hall 2nd floor, 310-338-4216) as early in the semester as possible. All discussions will remain confidential.

Please visit *http://www.lmu.edu/dss* for additional information. Ask for help as early in the semester as possible!

Also keep in mind that resources are available through the Library (*https://library.lmu.edu*) and Information Technology Services (*https://its.lmu.edu*). The DSS Office can help students connect with the appropriate person at the Library and ITS.

Topics and Important Dates

Correlated outcomes are shown for each topic. Specifics may change as the course progresses. University dates (italicized) are less likely to change.

August	Overview; introduction to data sets and database application tiers; hands-on with a database fiddle (*1a–1c*, *3c*)
September	File databases; database server setup, initialization, startup and shutdown, and general use; databases as managed services *(3a–3d)*; introduction to relational databases and SQL *(1b–1c)*
September 4	*Last day to add or drop a class without a grade of W*
October	Advanced SQL: aggregation, subqueries, JSON fields *(1c)*; document-centric databases *(1d, 2b, 3a–3d)*
November	Graph databases *(1d, 2b, 3a–3d)*; relational database theory: algebra and calculus; functional dependencies and normalization; integrity constraints *(2a)*
November 6	*Withdraw/credit/no-credit deadline*
November 25–27	*Thanksgiving; no class*
December	Database SDK work sessions *(1a–1d, 2b, 3a–3d, 4a–4e)*; miscellaneous topics *(time permitting)*
December 14	*Last set of term portfolio assignments due*

You can view my class calendar and office hour schedule in any iCalendar-savvy client. Its subscription link can be found on the course web site (it's too long to provide in writing).

Tentative Nature of the Syllabus

If necessary, this syllabus and its contents are subject to revision. Students are responsible for any changes or modifications announced or distributed in class, emailed to students' LMU Lion accounts, or posted on LMU's course management system, Brightspace. If you are absent from a synchronous class session, it is the student's responsibility to check Brightspace and with the professor to see if you missed any important class announcements. Students should not rely on word-of-mouth from classmates.

Course Evaluations

Student feedback provides valuable information for continued improvement. All students are expected to fairly and thoughtfully complete a course evaluation for this course. This semester, course evaluations will be administered online through the Blue™ evaluation system. You will receive an email notification at your Lion email address when the evaluation form is available. You may also access the evaluation form on Brightspace (*https://brightspace.lmu.edu*) dashboard during the evaluation period. Your responses will be anonymous and will not be linked to you in any way.

Course Outcomes

1 Know and understand how databases are designed, implemented, and deployed.

1a	*Know and understand the structure of modern database applications.*	Modern database applications include not only the database layer itself, but other layers for logic beyond pure data access.
1b	*Know and understand the relational database model.*	Outcomes *1b* and *1c* include the ability to interact directly with a "bare" database system—for this course, this is PostgreSQL. Relevant activities include creating tables with appropriate keys and foreign keys, loading these tables with data, performing a variety of queries, and altering a database schema.
1c	*Be proficient at database definition and manipulation with SQL.*	
1d	*Know about alternatives to the relational database model.*	For this course, this includes using files directly as a database as well as document-centric and graph databases.

2 Be acquainted with database system theory and algorithms.

2a	*Know the central concepts behind relational database theory.*	In addition to the data model itself, relational database theory includes the relational algebra, relational calculus, functional dependencies, normalization, and integrity constraints.
2b	*Be aware of the primary implementation and performance issues that database systems face.*	Issues include transaction management, security, storage, indexing, and query processing & optimization.

3 Apply this knowledge by designing and implementing a database application library for a particular domain.

3a	*Install, set up, and manage a database server.*	The intent of this outcome is to make you as comfortable with installing and running industrial-strength database servers as you are with more conventional types of applications.
3b	*Load data from flat files into a database server.*	These outcomes are demonstrated by writing code. Thus, some specific set of technologies, languages, and libraries unavoidably must be learned and used in order to accomplish these. However, it must also be understood that these concepts are general and technology-independent: when called for, one should be able to transfer this knowledge to other platforms.
3c	*Perform create, read, update, and delete operations (CRUD).*	
3d	*Design and implement a programming layer that will perform CRUD as function calls for a particular domain or set of use cases.*	

4 Follow disciplinary best practices throughout the course.

4a	*Write syntactically correct, functional code.*	Code has to compile. Code has to work. No errors, no bugs. Use unit tests as much as possible.
4b	*Use programming best practices, demonstrating principles such as DRY, proper separation of concerns, correct scoping of variables and functions, etc.*	This is the basis of good software design. It makes software easier to maintain, improve, and extend. Heed feedback well. *What you learn here will apply to future work in this field, in school and beyond.*
4c	*Write code that is easily understood by programmers other than yourself.*	This outcome involves all aspects of code readability and clarity for human beings, including but not limited to spacing & indentation, proper naming, presenting code in a manner that is consistent with its structure, documentation & comments when appropriate, and adherence to conventions or standards.
4d	*Use available resources and documentation to find required information.*	The need to look things up never goes away. Remember also that the course instructor counts as an "available resource," so this outcome includes asking questions and using office hours.
4e	*Use version control effectively.*	In addition to simply using version control correctly, effective use also involves appropriate time management, commit frequency, and descriptive commit messages.
4f	*Meet all designated deadlines.*	