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 relational database application that is accessible as a modern web service.

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

- Assorted handouts, articles, and sample code to be distributed throughout the semester.

The following texts can serve as programming tutorials and references:

- Bill Burke. RESTful Java with JAX-RS, O'Reilly Media, 2009.

In addition, do not hesitate to look for further information regarding the concepts, techniques, tools, and paradigms that we will discuss.
With great flexibility comes great accountability. First, you must submit your homework on time. The assignment due date is encoded in the homework number. Late homework detracts from outcome 4f (Meet all designated deadlines).

Quizzes and Tests

Some outcomes are best demonstrated by answering questions or doing exercises in class. These resemble traditional quizzes and tests, but, like homework, they are evaluated according to standards and do not produce a numerical score. They are typically spontaneous and unannounced.

Questions may include content-oriented elements as well as forward-looking, applicative portions (i.e., “use this knowledge to resolve this situation”). Tests are open-paper-everything; no sharing. “Open computer” might be allowed depending on the circumstances. You may neither solicit nor give help during a quiz or test. Late or missed tests are handled case-to-case; in all instances, talk to me.

Term Portfolio

Your accumulated homework and tests for the semester comprise the term portfolio—the final, definitive artifact that demonstrates the proficiencies you have reached for each course outcome. The term portfolio provides you with an opportunity to polish the work done throughout the semester; it is how you show that you learned from your mistakes or improved on already established knowledge.

Throughout the semester, you may improve your work based on received feedback and show it to me for re-evaluation. Improvements in proficiency are recorded and give you a good idea of how your term portfolio will fare long before its final version is submitted.

The final version of the term portfolio is due on December 14. Late portfolios detract from outcome 4f.

Extra Credit

In terms of standards-based grading, “extra credit” takes on a different meaning: it indicates work that, if successfully performed, would indicate advanced proficiency (+). Extra credit tasks may be assigned for homework, quizzes/tests, or the final term portfolio. Accomplish them successfully to rack up those +’s. You do not need to perform extra credit work to show advanced proficiency; it merely demonstrates such proficiency more readily.

Version Control

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

Attendance

Attendance at all sessions is expected, but not absolutely required. If you must miss class, it is your responsibility to keep up with the course. The last day to add or drop a class without a grade of W is August 31. The withdrawal or credit/no-credit deadline is November 2.

University Policy on Academic Honesty

Loyola Marymount University expects high standards of honesty and integrity from all members of its community. All students are expected to follow the LMU Honor Code and Process, as stated in the LMU Undergraduate Bulletin.

Americans with Disabilities Act

Students with special needs as addressed by the Americans with Disabilities Act who need 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 (physical, learning, or psychological) needing academic accommodations should contact DSS (Daum 224, x84535) as early in the semester as possible. All discussions will remain confidential. Please visit http://www.lmu.edu/dss for additional information.
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
- Working with databases and RESTful web services—setup, initialization, startup and shutdown, general use (1a, 3a, 3d); version control setup (4e)

August 31
- Last day to add or drop a class without a grade of W

September
- Database application tiers (1a, 3a–3d); the relational model (1b); SQL (1c); database design and definition (1b, 1c); object-relational mapping (3b); database and web service unit tests (4a)

October
- Relational database theory: algebra and calculus (2a); functional dependencies and normalization (2a); integrity constraints (2a)

November
- Database implementation issues: transactions, security, storage, indexing, query processing and optimization (2b); big data and non-relational databases (“NoSQL”) (1d)

November 2
- Withdraw/credit/no-credit deadline

November 21–23
- Thanksgiving; no class

December
- Portfolio improvement workshops (1a–4e); miscellaneous topics (varies)

December 14
- Term portfolios 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).

If necessary, this syllabus and its contents are subject to revision. Students are responsible for any changes or modifications announced in class.
## 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 a service layer 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 emerging alternatives to the relational database model. The extent of the expected knowledge will be determined by available time. Planned coverage focuses on big data and NoSQL.

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 relational database application that is accessible as a modern web service.**

   - **3a** Install, set up, and manage a PostgreSQL server. The intent of this outcome is to make you as comfortable with installing and running this industrial-strength database server as you are with more conventional types of applications.
   - **3b** Implement a data model using Java Persistence annotations. These outcomes are demonstrated by writing code that involves one or more of these areas. Thus, some specific set of technologies, languages, and libraries must be learned and used—for this course, we focus on Java-based technologies accessed through web protocols. 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** Design and implement data access objects.
   - **3d** Design and implement RESTful web services served by Apache Tomcat and built on the Jersey library.

4. **Follow academic and technical best practices throughout the course.**

   - **4a** Write syntactically correct, functional code. Code has to compile. Code has to work. No errors, no bugs. Database and web service unit tests must be supplied.
   - **4b** Demonstrate proper separation of concerns. This is the basis of good software design. It makes software easier to maintain, improve, and extend. Proper separation of concerns includes but is not limited to correct scoping of variables & functions and zero duplication of code.
   - **4c** Write code that is easily understood by programmers other than yourself. This outcome involves all areas of code readability and clarity, including but not limited to documentation & comments, spacing & indentation, proper naming, 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 commit frequency and descriptive commit messages.
   - **4f** Meet all designated deadlines.
### Sample Standards Achievement Report

Based on these proficiencies, the student will get a C+.

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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>Know and understand the structure of modern database applications.</td>
</tr>
<tr>
<td>1b</td>
<td>Know and understand the relational database model. +</td>
</tr>
<tr>
<td>1c</td>
<td>Be proficient at database definition and manipulation with SQL. +</td>
</tr>
<tr>
<td>1d</td>
<td>Know about emerging alternatives to the relational database model.</td>
</tr>
</tbody>
</table>

2. **Be acquainted with database system theory and algorithms.**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2a</td>
<td>Know the central concepts behind relational database theory. /</td>
</tr>
<tr>
<td>2b</td>
<td>Be aware of the primary implementation and performance issues that database systems face. /</td>
</tr>
</tbody>
</table>

3. **Apply this knowledge by designing and implementing a relational database application that is accessible as a modern web service.**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3a</td>
<td>Install, set up, and manage a PostgreSQL server. +</td>
</tr>
<tr>
<td>3b</td>
<td>Implement a data model using Java Persistence annotations. +</td>
</tr>
<tr>
<td>3c</td>
<td>Design and implement data access objects. +</td>
</tr>
<tr>
<td>3d</td>
<td>Design and implement RESTful web services served by Apache Tomcat and built on the Jersey library. +</td>
</tr>
</tbody>
</table>

4. **Follow academic and technical best practices throughout the course.**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4a</td>
<td>Write syntactically correct, functional code. +</td>
</tr>
<tr>
<td>4b</td>
<td>Demonstrate proper separation of concerns, especially MVC. +</td>
</tr>
<tr>
<td>4c</td>
<td>Write code that is easily understood by programmers other than yourself. +</td>
</tr>
<tr>
<td>4d</td>
<td>Use available resources and documentation to find required information.</td>
</tr>
<tr>
<td>4e</td>
<td>Use version control effectively. /</td>
</tr>
<tr>
<td>4f</td>
<td>Meet all designated deadlines. /</td>
</tr>
</tbody>
</table>