CS 361 – Database Systems
Spring 2018

Instructor: George Thomas **Office:** Halsey 218
Email: thomasg@uwosh.edu **Phone:** 424-2069

Office Hours:
MW 11:30-1:00
T 11:30-12:30
R 8:30-9:30
Or **by appointment**

Lectures: TR 1:20-2:50, Halsey 202

Prerequisites: CompSci-271 with a grade of C of better.

Recommended Textbooks:

- *Database Systems: The Complete Book* (2nd Edition), Hector Garcia-Molina, Jeffrey D. Ullman and Jennifer Widom, Prentice Hall, 2008
- *Database System Concepts* (6th Edition), Abraham Silberschatz, Henry Korth and S. Sudarshan, McGraw-Hill, 2010

Course Website: UWO D2L

Note: If you have special needs, please come and talk to me at the end of the first class.

Course Information

An introduction to database processing with emphasis on database techniques, design and modeling. Programming projects may include implementation of selected database processing methods and the use of database software.

Topic Coverage

- Overview of database systems and introduction to database design
- The relational model, relational algebra and calculus
- Schema refinement and normal forms
- SQL: queries, constraints, triggers
- Database application development
- Overview of storage and indexing
- Overview of query evaluation and query optimization
- Overview of transaction management, concurrency control and crash recovery
- Non-relational Databases – XML, MongoDB, JSON, etc.

Learning Outcomes

Given our coverage of these topics, by the end of the course, you will be expected to know the following:

- Fundamental concepts of database design
  - model customer requirements of a relational database with an Entity-Relational diagram (E-R diagram)
  - transform an E-R diagram to a database schema
  - write a query to a relational database in SQL
  - formulate a query to a relational database from the basic operators in relational algebra
  - design a database to provide the necessary information for an organization while minimizing redundancy and null entries.
- SQL
  - design and create a database using the Data Definition Language of a Database Management System
  - write queries to a relational database in an interactive mode
- Design of "good" relations, Schema Refinement, Concept of normalization and other theoretical issues
  - formulate the integrity constraints in the form of functional dependencies
  - eliminate extraneous attributes in a functional dependency
  - eliminate redundant functional dependencies
  - develop a cover from a set of functional dependencies
  - evaluate a proposed relational schema and determine whether it is in Third-Normal-Form (3NF) or Boyce-Codd-Normal-Form (BCNF)
  - implement a normalization program that checks whether a proposed relational schema is in 3NF or BCNF
  - decompose proposed relational schemas that are not in 3NF or BCNF into 3NF or BCNF
- Basic file organization and various file structure methods
  - determine the access time of records based on the file organization and file structure
  - specify the type of stable and non-stable storage in the design of a database management system.
  - analyze the requirements and select the design of an index (Hash, B+ tree)
  - organize data on disk to minimize disk accesses for various queries.
- Algorithms and implementation of large database systems
  - analyze the need of a database operator (scan, equality search, range search, insert, delete etc) and determine an appropriate and/or efficient algorithm (external sorting, hash, B+, clustered vs. unclustered, various join algorithms) in its implementation.
  - implement a Relational Operation (Example: Join)
- Transaction processing, concurrency issues, and recovery
  - identify and prevent deadlocks in concurrent database accesses
  - be able to describe the recovery process of databases
  - design the data structure and program of a database recovery mechanism

- provide accurate, consistent, and efficient transactions within the context of concurrency issues and the possibility of various kinds of failures, such as a system crash.

Course Grading Policy: Your final grade for this course will be based on four components, namely Exams, Homework, Programming and Modeling Assignments, and general class participation. Your overall numerical grade for the course will be computed as the weighted sum of the component grades using the following weights:

Component	Weight
Exams (3)	45%
Homework	20%
Programming and Modeling Assignments	30%
Class participation	5%

Tentative Exam Dates are as follows:

- **Exam 1 – Thursday, 2/22**
- **Exam 2 – Thursday, 4/05**
- **Exam 3 – Thursday, 5/10**

Your letter grade for the course will be computed as follows:

Numerical Score	Grade	Numerical Score	Grade
>=92	A	72-78	C
90-92	A-	70-72	C-
88-90	B+	68-70	D+
82-88	B	62-68	D
80-82	B-	60-62	D-
78-80	C+	<60	F

While this overall grading scheme is fixed, I will be happy to discuss any issue you may have with individual grades. If you notice a mistake or have a question regarding a specific grade, please come and talk to me *as soon as possible*. Do not wait until the end of the semester to bring up grading issues.

Attendance and Participation: You are expected to not only attend every class meeting on time but also to come prepared for and participate actively in it. Necessary preparation requires you to have studied and assimilated the material covered in previous sessions, to have met with me outside of class to discuss any questions you may have, to have done any assigned readings, homework or assignments on time. I **strongly encourage you to ask any question** or raise any issue you have with the course either during or at the end of class, or during my office hours. I will also gladly meet with you by appointment.

Deadlines: Each homework will come with a deadline by which it must be submitted (usually, at the start of a lecture). **Late homework submissions will NOT be accepted**.

Each programming or modeling assignment will also come with a deadline (day and time) by which it must be submitted. You are allotted *three* credit days you can use through the semester. A credit day is exactly 24 hours or less. You can use unused credit days to submit an assignment after its deadline, without penalty. Any assignment submitted after the deadline, plus any credit days you have unused, will receive a zero.

For example, if you have 2 unused credit days available and a assignment is due on Tuesday at 5:00PM, you can submit it anytime by exactly Thursday at 5:00PM without penalty. Do note that if you submit your assignment on Thursday at 5:01PM, you will be penalized 100% of the score of the project and thus receive a zero! Note also that if you submit your assignment on Wednesday at 5:01PM, you will be charged two credit days (but no penalty, obviously).

Extensions and Absences: Extensions on deadlines may be granted at the discretion of the instructor if you provide a valid justification (in the form of a written excuse from a medical doctor or the Dean of Students Office) **before** the due date.

If you miss a scheduled exam (tentative dates are provided), you **may** be able to take a make-up exam provided you give the instructor a valid justification (see above) ahead of time if possible. Only one make-up exam will be given. It will be a comprehensive exam scheduled at the end of the semester.

Collaborating versus Cheating: Unless otherwise stated in the assignment or project, all submissions must be entirely your own work. While it is acceptable to discuss the assignments at a high level (for example, at the design level) with others, you must submit your own work. **You may not "borrow" any piece of code or design of any length from someone else, the internet, or any other source, unless you can live with a zero and the other potential academic sanctions of cheating** (see UWO Student Discipline Code 2007, Chapter UWS 14).