TENNESSEE TECH UNIVERSITY
COMPUTER SCIENCE
CSC 3300 DATABASE SYSTEMS
SECTION 002, MONDAYS, WEDNESDAYS, AND FRIDAYS 1115AM-1210PM,
FNDH 330
3 CREDIT HOURS, SPRING 2020

INSTRUCTOR INFORMATION

Instructor's Name: Dr. Michael D. Rogers
Office: FNDH 134
Telephone Number: No phone available
Email: mrogers@tntech.edu
OFFICE HOURS: TBA

COURSE INFORMATION

Prerequisites (if applicable): Junior Standing and C or better in CSC 1610, CSC 2110, and CSC 2111.
Required: *Database System Concepts 6th Ed.* Silberschatz, Korth, and Sudarshan

References (if applicable): None

COURSE DESCRIPTION: Organization and management of large data files; data definition; database models; query languages; crash recovery; concurrency control; and case studies.

COURSE OBJECTIVES/STUDENT LEARNING OUTCOMES

1. The basic concepts of relational theory, with techniques for relational database design.
2. Skills for modeling enterprise information.
3. Querying and manipulating relational data using a SQL.
4. Querying and manipulating relational data using a general purpose programming language.
5. Problem analysis, design, and implementation of a complete relational database application.
6. Basic security issues including access controls and data integrity.

MAJOR TEACHING METHODS

Teaching methods may include: lectures, labs, demonstrations, discussion, reading, written assignments, and programming assignments.

SPECIAL INSTRUCTIONAL PLATFORM/MATERIALS

Moodle is used to deliver material for the course. (http://moodle.csc.tntech.edu/

TOPICS TO BE COVERED

- Chapter 1. An overview of database management.
- Chapter 2. An introduction to relational databases.

- Chapter 3. An Introduction to SQL. Introduction to MySQL Language bindings to MySQL
- Chapter 4. Intermediate SQL.
- Chapter 5. Advanced SQL
- SecKnitKit: Database vulnerabilities and attacks. Database security, access controls, and integrity.
- Chapter 6. Formal Languages. Tuples, Relational Algebra, Relation Calculus
- Chapter 7. The Entity Relationship Model.
- Chapter 8. Database Design. Functional Dependencies, Normalization.
- Chapters 11 and 12. Functional Dependencies. Normalization.

Grading and Evaluation Procedures

The grading for the course is based on the following components:

Component	Weight
Homework	40%
Quizzes/Exams	60%
Rounding Quiz	+0.5%

GRADING SCALE

Letter Grade	Grade Range
A	90-100
B	80-89
C	70-79
D	60-69
F	60

COURSE POLICIES

STUDENT ACADEMIC MISCONDUCT POLICY

Maintaining high standards of academic integrity in every class at Tennessee Tech is critical to the reputation of Tennessee Tech, its students, alumni, and the employers of Tennessee Tech graduates. The Student Academic Misconduct Policy describes the definitions of academic misconduct and policies and procedures for addressing Academic Misconduct at Tennessee Tech. For details, view the Tennessee Tech's Policy 217 – Student Academic Misconduct at Policy Central (https://goo.gl/KD49cp)

Cheating, in particular plagiarism, will result in an automatic failure. Just don't do it. Faking program output counts as cheating! The first time you are caught faking output on a programming assignment, you will receive a 0 grade for that assignment. If you are caught again, it will result in automatic failure.

Any student who violates the university's, department's, or class's academic honesty policy will be reported to the university authority.

Attendance Policy

The professor may decide to take roll for the semester. If the professor does take roll and you miss four or more hours, then your total grade will be reduced by 2^{m-1} points where m is the number of hours missed.

Please turn off your cell phones or at least put them in silent mode. Don't use your laptop during the class unless instructor asks you to.

Assignments and Related Policy

Your instructor will never accept assignments, or part of assignments, of any type via email for any reason!

All assignments are due at the date assigned during class. More than enough time is given to complete assignments, and therefore exceptions will not be made for any reason, including lab down-time or assignments due in other classes on the same due date. Each day a programming assignment is late is penalized $5(2^{x-1})$ points, where x is the number of days late. Saturdays and Sundays count as late days. No other type of assignment will be accepted late, no exceptions.

The professor will NOT regrade assignment that you have turned in erroneously, e.g. you forget to commit your submission, you turn in a binary executable instead of the source code, etc. The submission procedure allows you to download your submission and check it, so there no excuse for an improper submission. Also, your computer crashing or having network problems is also no excuse.

Additionally, you will not be allowed to make up tests and quizzes that you miss, barring exceptional circumstances (for example, hospitalization) at the discretion of the instructor. If you do not finish a test/quiz before the allotted time, un-submitted questions will be graded as 0. You will not be given another opportunity to answer missed questions.

The professor may indicate that an assignment is to be done using a particular programming language, operating system, code library, or document typesetting system. An assignment that does not meet the professor's explicit constraints will receive a grade of zero. Similarly, the professor may indicate during class extra constraints or clarifications that are not a part of the assignment's original document (either web page or handout). Not adhering to these additions will effect your grade, possibly resulting in zero grade.

Your programming assignments should reflect good programming practices. For example, you should use the appropriate data structures and their implementations should be efficient. Examples of poor programming practices include implementing order n appends and pop_front operations for a data structure that supports constant time append and pop_front, spaghetti code (particularly if you are using assembly language), and little or no thought for design (e.g. not collecting like functions into modules/classes). Poor programming practices will result in a reduction in your grade.

For a programming assignment, the instructor may ask you to create a makefile. The make utility will use your makefile to automatically compile and run your applications. It is your responsibility to become familiar with the make utility and makefiles (this utility is standard on Unix systems and can be found on MS Windows systems having a C/C++ compiler installed). If your program does not compile, the professor will assign you a grade of zero for the assignment. Make sure your program compiles (test it) before you turn it in! Similarly, you are responsible for fully testing your assignment before you turn it in. Students often believe that their assignment works (with little or no testing), only to find that the professor uncovers major programming flaws because he has a slightly different environment or uses more/different test cases. It is important to note that the professor may test your program in

a variety of ways, including using different main drivers than the one you turn in (in the case of libraries), or even modifying your existing code.

Office Etiquette:

If the professor's office door is open (even just a crack), that means that the professor is willing to take questions and help students. The professor is often available outside of his normal office hours. You may ask the professor if you can stop by his office during certain hours that are not normal office hours.

Do not use office hours or appointments as an opportunity to demo your assignment, which is akin to asking your professor to grade your assignment twice.

It is very rude to enter the professor's office when the professor is helping another student. Do not enter the professor's office unless you are invited.

Do not write code in the professor's office, unless you are explicitly asked to do so by the professor. Some students wish to sit in the professor's office and have the professor fix each of their problems as they code. Often, student exhibit this behavior because they have waited too late and need their professor to do their assignment for them so that they can turn it in on time. However, you learn nothing unless you spend serious time working through difficulties yourself.

Do not sit outside the professor's office door coding. Students often sit outside the professor's office so that they can quickly run in his office and get him to fix their bugs. See the above paragraph. A good rule of thumb is the following: if you come to the professor's office more that twice in a single day for help on a particular programming assignment, you aren't spending enough time trying to work through your problems yourself.

Disability Accommodation

Students with a disability requiring accommodations should contact the Office of Disability Services (ODS). An Accommodation Request (AR) should be completed as soon as possible, preferably by the end of the first week of the course. The ODS is located in the Roaden University Center, Room 112; phone 372-6119. For details, view the Tennessee Tech's Policy 340 – Services for Students with Disabilities at Policy Central (https://goo.gl/aC0qOa)

Notes

The instructor reserves the right to modify this syllabus as necessary to accommodate course needs.