**Course Description:**

Databases management systems are a crucial part of most large-scale industry and open-source systems. This course provides comprehensive coverage of issues associated with database system development and an in-depth examination of structures and techniques used in contemporary database management systems (DBMSs). Students will learn about the inner workings of these exciting systems: Which algorithms are used? What are typical architectures used to build a system as complex as a DBMS? What are implementation strategies? These questions and more will be answered during the course.

The course is highly applied, emphasizing practical skills and habits through a series of programming assignments during which students will develop their own tiny DBMS like engine. We will cover the most important aspects/components of a DBMS: storage and buffer **management, indexing, query optimization, query execution, and concurrency control and recovery.**

**Course Material:**

The following text books will be helpful for following the course and studying the presented material. All four textbooks have their merits, but any one should be su\_cient as reading material.

* Garcia-Molina, Ullman, and Widom, **Database Systems: The Complete Book, 2nd Edition**, Prentice Hall, 2008
* Elmasri and Navathe , **Fundamentals of Database Systems, 6th Edition**, Addison-Wesley , 2003
* Ramakrishnan and Gehrke , **Database Management Systems, 3nd Edition**, McGraw-Hill , 2002
* Silberschatz, Korth, and Sudarshan , **Database System Concepts, 6th Edition**, McGraw Hill , 2010

The slides will be made available on blackboard every week.

**Prerequisites:**

* Courses:CS425
* Programming experience in C, C++ or other low level languages
* Unix OS

**Course Details:**

The following topics will be covered in the course:

**1. Introduction**

* Relational Algebra
* DBMS Architecture

**2. Hardware Characteristics affecting DBMS Design**

* Read/Write Properties of Disks
* RAID Storage
* Memory Hierarchy

**3. Disk Storage and Buffer Management**

* Physical Tuple Layout
* Page Layout
* Tuple IDs
* Buffer Replacement Strategies

**4. Indexing and Hashing**

* B-Tree-Family Indices
* Hashing

**5. Query Optimization**

* Logical Optimization
* Equivalence Transformations
* Physical Optimization
* Join Reordering
* Cost Estimation

**6. Query Execution**

* Pipelining
* Push vs. Pull based Execution
* Access Methods
* Join Methods
* Grouping and Aggregation
* Other Operator Implementations
* External Sorting

**7. Recovery**

* Write Ahead Log (WAL)
* Algorithms for Recovery and Isolation Exploiting Semantics (ARIES)

**8. Concurrency Control**

* Serializability
* Two-Phased Locking (2PL)
* Implementing of Locks

**9. Advanced Topics / optional**

* Distributed Database Systems
* Data warehousing
* Parallel Query Execution
* Technics for Executing Nested Queries and Un-nesting
* Additional Index Structures
* Relation to Large-Scale Data Analytics

**Workload and Grading Policies:**

**Programming Assignments:**

There will be several programming assignments during the course. Starting from a storage manager you will be implementing your own tiny database-like system from scratch. You will explore how to implement the concepts and data structures discussed in the lectures and readings. The assignments will require the use of skills learned in this course as well as other skills you have developed throughout your program. Each assignment will build upon the code developed during the previous assignment. In the end there will be an optional assignment for extra credit. Each of the regular assignments will have optional parts that give extra credit. All assignments have to be implemented using C/C++.

**\_ Assignment 1 -** Storage Manager: Implement a storage manager that allows read/writing of blocks to/from a file on disk

**\_ Assignment 2** - Buffer Manager: Implement a buffer manager that manages a buffer of blocks in memory including reading/cashing to disk and block replacement (cashing blocks to disk to make space for reading new blocks from disk).

**\_ Assignment 3** - Record Manager: Implement a simple record manager that allows navigation through records, and inserting and deleting records.

**\_ Assignment 4** - B+-Tree Index: Implement a disk-based B+-tree index structure.

**\_ Potential Optional Assignment:**

* Implement a standard operator algorithm on top of your record manager, e.g., nested loop join, hash-aggregate, . . . .

**Mid Term and Final Exam:**

There will be a midterm and a final exam covering the topics of the course.

**Grading Policies:**

\_ Programming Assignments: 40% (10% + 10% + 10% + 10%)

\_ Mid Term Exam: 25%

\_ Final Exam: 30%

\_ other: 5%

**Course Objectives:**

After attending the course students should:

\_ Understand the design decisions behind textbook DBMS architectures

\_ Know the trade-offs of various storage organization techniques

\_ Be able to build parts of a small-sized data processing system from scratch

\_ Understand the basics of query optimization

\_ Know standard implementations of relational operators such as join, aggregation, and set operations

\_ Be able to estimate the cost of executing an operator/query based on DB statistics

\_ Know standard database indexing techniques

\_ Understand concurrency control and recovery mechanisms