Architecture and Implementation of Database Management Systems

Prof. Dr. Marc H. Scholl

Winter 2006/07
University of Konstanz, Dept. of Computer & Information Science

www.inf.uni-konstanz.de/dbis/teaching/
Module 1: Introduction & Overview

Module Outline

1.1 What it’s all about
1.2 Outline of the Course
1.3 Organizational Matters
1.1 What it’s all about

While introductory courses have presented the functionality, i.e., the interface, of database management systems (DBMSs), this course will dive into the internals. We will, for example, learn how a DBMS can

- efficiently organize and access data on disk, knowing that I/O is way more expensive than CPU cycles,
- translate SQL queries into efficient execution plans, including query rewrite optimization and index exploitation,
- sort/combine/filter large data volumes exceeding main memory size by far,
- allow many users to consistently access and modify the database at the same time,
- takes care of failures and guarantees recovery into a consistent operation after crashes.

This is a systems-oriented course, with focus on the necessary infrastructure to build a DBMS. This will help to thoroughly analyze, compare, and tune DBMSs for performance-critical applications.
1.1.1 Overall System Architecture

A DBMS is typically run as a back-end server in a (local or global) network, offering services to clients directly or to application servers.

Generally, we call this the “3-tier reference architecture”.
1.1.2 Layered DBMS Architecture

Typically, a DBMS implements its functionality in a layered architecture that builds up by incrementally adding more abstractions from the low level of block I/O devices up to the high level of a declarative (SQL) user interface.
Whether the DBMS offers relational, object-relational, or other data structures at the user interface, *internally* they have to be mapped into fixed-length blocks that serve as the basic I/O-unit of transfer between main and secondary memory.
1.1.4 Access Paths

A DBMS typically provides a number of indexing techniques that allow for fast content-based searching of records, such as tree-structured or hash-based methods. Often, the suite of such indexing techniques can be extended to match the requirements of particular applications.
1.1.5 Query Execution

Declarative query specifications, e.g. expressed in SQL, need to be optimized and transformed into efficient query execution plans (QEPs), i.e., sequential or even parallelized programs that compute the results.
1.1.6 Implementing a Lock Manager

Most DBMSs use a locking protocol (e.g., 2PL) for concurrency control. Efficiently implementing the lock manager and exploiting the synchronization primitives offered by the underlying operating system is crucial for a high degree of parallelism.
1.2 Outline of the Course

We will pursue a bottom-up strategy, starting from the block-I/O devices used for secondary storage management and work our way up to the SQL interface.

Most part of the lecture is based on the book (Ramakrishnan and Gehrke, 2003). Additional references to other textbooks and related literature will be given when appropriate.
1.3 Organizational Matters

▶ Register with the Account Tool.

▶ *Actively* participate in lectures and assignments.

▶ There will be a written exam at the end of the semester.

▶ Let us know when you have problems or suggestions.

▶ 10 copies of the book underlying this course are available in the U KN library.
Bibliography


