

Introduction to CS 4604

T. M. Murali

August 24, 2009

Course Information

▶ Instructor

- ▶ T. M. Murali, 2160B Torgerson, 231-8534, murali@cs.vt.edu
- ▶ Office Hours: 9:30am–11:30am Mondays and Wednesdays

▶ Teaching assistant

- ▶ M. Shahriar Hossain, msh@cs.vt.edu
- ▶ Office Hours: 12pm-2pm, Tuesdays and Thursdays, McBryde 106/110.

▶ Class meeting time

- ▶ 4pm–5:15pm, Mondays and Wednesdays, 307 McBryde

▶ Keeping in Touch

- ▶ Course web site <http://courses.cs.vt.edu/~cs4604>, **updated regularly through the semester**
- ▶ Listserv: cs4604_91850@listserv.vt.edu

▶ Prerequisites: a grade of C or better in CS 2604, senior standing

Textbook

▶ Required

- ▶ *A First Course in Database Systems*, **Third edition**, Ullman and Widom, Prentice Hall, 2008.
- ▶ Web page for the book
<http://www-db.stanford.edu/~ullman/fcdb.html>
- ▶ Also consult list of errata on this web page.

Course Grading

Homeworks	30%	5–6
Midterm exam	15%	October 14
Final exam	25%	December 11
Course project	30%	7 assignments

- ▶ Projects and homework assignments alternate.
- ▶ Submit hard copies of homeworks and project assignments at the start of class on the due date.
- ▶ Each class has required reading. Please consult the course web page.

Course Project

- ▶ Project overview <http://courses.cs.vt.edu/~cs4604/Fall109/project/project.html>
- ▶ 1, 2, or 3 persons per project.
- ▶ Project runs the entire semester with regular assignments and a final implementation assignment.
- ▶ You are free to suggest a project. Talk to me first!
- ▶ Send email to Shahriar by 5pm Friday, Aug 28, 2009 stating which project you want to work on.
 - ▶ No more than two groups per project topic.
 - ▶ Check webpage to see which projects are free.

Why Study Databases?

- ▶ Academic:
 - ▶ Databases involve many aspects of computer science
 - ▶ Fertile area of research
 - ▶ Three Turing awards in databases

Why Study Databases?

- ▶ Academic:
 - ▶ Databases involve many aspects of computer science
 - ▶ Fertile area of research
 - ▶ Three Turing awards in databases
- ▶ Programmer: a plethora of applications involve using and accessing databases

Why Study Databases?

- ▶ Academic:
 - ▶ Databases involve many aspects of computer science
 - ▶ Fertile area of research
 - ▶ Three Turing awards in databases
- ▶ Programmer: a plethora of applications involve using and accessing databases
- ▶ Businessman: Everybody needs databases \Rightarrow lots of money to be made.

Why Study Databases?

- ▶ Academic:
 - ▶ Databases involve many aspects of computer science
 - ▶ Fertile area of research
 - ▶ Three Turing awards in databases
- ▶ Programmer: a plethora of applications involve using and accessing databases
- ▶ Businessman: Everybody needs databases \Rightarrow lots of money to be made.
- ▶ Student:

Why Study Databases?

- ▶ Academic:
 - ▶ Databases involve many aspects of computer science
 - ▶ Fertile area of research
 - ▶ Three Turing awards in databases
- ▶ Programmer: a plethora of applications involve using and accessing databases
- ▶ Businessman: Everybody needs databases \Rightarrow lots of money to be made.
- ▶ Student:
 - ▶ Get those last three credits and I don't have to come back to Blacksburg ever again!!!
 - ▶ Google will hire me!!
 - ▶ Databases sound cool!
 - ▶ ???

What Will You Learn in CS4604?

- ▶ Implementation
 - ▶ How do you build a system such as ORACLE or MySQL?
- ▶ Programming
 - ▶ How do you use the capabilities of a DBMS?
- ▶ Design
 - ▶ How do you model your data and structure your information in a database?

What Will You Learn in CS4604?

- ▶ Implementation
 - ▶ How do you build a system such as ORACLE or MySQL?
- ▶ Programming
 - ▶ How do you use the capabilities of a DBMS?
- ▶ Design
 - ▶ How do you model your data and structure your information in a database?
- ▶ CS 4604 achieves a balance between

What Will You Learn in CS4604?

- ▶ Implementation
 - ▶ How do you build a system such as ORACLE or MySQL?
- ▶ Programming
 - ▶ How do you use the capabilities of a DBMS?
- ▶ Design
 - ▶ How do you model your data and structure your information in a database?
- ▶ CS 4604 achieves a balance between
 - ▶ creating, querying, and implementing realistic databases and

What Will You Learn in CS4604?

- ▶ Implementation
 - ▶ How do you build a system such as ORACLE or MySQL?
- ▶ Programming
 - ▶ How do you use the capabilities of a DBMS?
- ▶ Design
 - ▶ How do you model your data and structure your information in a database?
- ▶ CS 4604 achieves a balance between
 - ▶ creating, querying, and implementing realistic databases and
 - ▶ a firm theoretical foundation to designing moderate-sized databases.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.
- ▶ Create E/R models from application descriptions.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.
- ▶ Create E/R models from application descriptions.
- ▶ Convert E/R models into relational designs.
- ▶ Identify redundancies in designs and remove them using normalisation techniques.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.
- ▶ Create E/R models from application descriptions.
- ▶ Convert E/R models into relational designs.
- ▶ Identify redundancies in designs and remove them using normalisation techniques.
- ▶ Create databases in an RDBMS and enforce data integrity constraints using SQL.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.
- ▶ Create E/R models from application descriptions.
- ▶ Convert E/R models into relational designs.
- ▶ Identify redundancies in designs and remove them using normalisation techniques.
- ▶ Create databases in an RDBMS and enforce data integrity constraints using SQL.
- ▶ Write sophisticated database queries using SQL.
- ▶ Understand tradeoffs between different ways of phrasing the same query.

Course Goals and Outcomes

- ▶ Take an English language description and convert it into a working database application.
- ▶ Create E/R models from application descriptions.
- ▶ Convert E/R models into relational designs.
- ▶ Identify redundancies in designs and remove them using normalisation techniques.
- ▶ Create databases in an RDBMS and enforce data integrity constraints using SQL.
- ▶ Write sophisticated database queries using SQL.
- ▶ Understand tradeoffs between different ways of phrasing the same query.
- ▶ Implement a web interface to a database.

Course Outline

- ▶ Weeks 1–5, 13: Query/Manipulation Languages
 - ▶ The relational model
 - ▶ Relational Algebra
 - ▶ SQL
 - ▶ Data definition
 - ▶ Programming with SQL
- ▶ Weeks 6–8: Data Modelling
 - ▶ Entity-Relationship (E/R) approach
 - ▶ Specifying Constraints
 - ▶ Good E/R design

Course Outline

- ▶ Weeks 1–5, 13: Query/Manipulation Languages
 - ▶ The relational model
 - ▶ Relational Algebra
 - ▶ SQL
 - ▶ Data definition
 - ▶ Programming with SQL
- ▶ Weeks 6–8: Data Modelling
 - ▶ Entity-Relationship (E/R) approach
 - ▶ Specifying Constraints
 - ▶ Good E/R design
- ▶ Weeks 9–13: Relational Design
 - ▶ Converting ER to “R”
 - ▶ Normalisation to avoid redundancy

Course Outline

- ▶ Weeks 1–5, 13: Query/Manipulation Languages
 - ▶ The relational model
 - ▶ Relational Algebra
 - ▶ SQL
 - ▶ Data definition
 - ▶ Programming with SQL
- ▶ Weeks 6–8: Data Modelling
 - ▶ Entity-Relationship (E/R) approach
 - ▶ Specifying Constraints
 - ▶ Good E/R design
- ▶ Weeks 9–13: Relational Design
 - ▶ Converting ER to “R”
 - ▶ Normalisation to avoid redundancy
- ▶ Week 14–15: Students’ choice
 - ▶ XML
 - ▶ Query optimisation
 - ▶ Data mining

What is a DBMS?

- ▶ Database:

What is a DBMS?

- ▶ Database: collection of data that exists for a long period of time.
- ▶ Database Management System (DBMS) = database + set of programs to access/manipulate it

What is a DBMS?

- ▶ Database: collection of data that exists for a long period of time.
- ▶ Database Management System (DBMS) = database + set of programs to access/manipulate it
- ▶ Features of a DBMS

What is a DBMS?

- ▶ Database: collection of data that exists for a long period of time.
- ▶ Database Management System (DBMS) = database + set of programs to access/manipulate it
- ▶ Features of a DBMS
 - ▶ Support massive amounts of data
 - ▶ Persistent and durable storage
 - ▶ Efficient and convenient access (create and query)
 - ▶ Secure, concurrent, and atomic access

What is a DBMS?

- ▶ Database: collection of data that exists for a long period of time.
- ▶ Database Management System (DBMS) = database + set of programs to access/manipulate it
- ▶ Features of a DBMS
 - ▶ Support massive amounts of data
 - ▶ Persistent and durable storage
 - ▶ Efficient and convenient access (create and query)
 - ▶ Secure, concurrent, and atomic access
- ▶ Examples?

What is a DBMS?

- ▶ Database: collection of data that exists for a long period of time.
- ▶ Database Management System (DBMS) = database + set of programs to access/manipulate it
- ▶ Features of a DBMS
 - ▶ Support massive amounts of data
 - ▶ Persistent and durable storage
 - ▶ Efficient and convenient access (create and query)
 - ▶ Secure, concurrent, and atomic access
- ▶ Examples?
- ▶ Search engines, banking systems, airline reservations, corporate records, payrolls, sales inventories.
- ▶ New applications: Wikis, biological/multimedia/scientific/geographic data, heterogeneous data.

Features of a DBMS

- ▶ Support **massive** amounts of data
 - ▶ Giga/tera/petabytes
 - ▶ Far too big for main memory

Features of a DBMS

- ▶ Support **massive** amounts of data
 - ▶ Giga/tera/petabytes
 - ▶ Far too big for main memory
 - ▶ Recent trend: databases run on single computers.

Features of a DBMS

- ▶ Support **massive** amounts of data
 - ▶ Giga/tera/petabytes
 - ▶ Far too big for main memory
 - ▶ Recent trend: databases run on single computers.
- ▶ **Persistent** storage
 - ▶ Programmes update, query, manipulate data.
 - ▶ Data continues to live long after programme finishes.

Features of a DBMS

- ▶ Support **massive** amounts of data
 - ▶ Giga/tera/petabytes
 - ▶ Far too big for main memory
 - ▶ Recent trend: databases run on single computers.
- ▶ **Persistent** storage
 - ▶ Programmes update, query, manipulate data.
 - ▶ Data continues to live long after programme finishes.
- ▶ **Efficient** and **convenient** access
 - ▶ Efficient: do not search entire database to answer a query.
 - ▶ Convenient: allow users to create and query the data as easily as possible.

Features of a DBMS

- ▶ Support **massive** amounts of data
 - ▶ Giga/tera/petabytes
 - ▶ Far too big for main memory
 - ▶ Recent trend: databases run on single computers.
- ▶ **Persistent** storage
 - ▶ Programmes update, query, manipulate data.
 - ▶ Data continues to live long after programme finishes.
- ▶ **Efficient** and **convenient** access
 - ▶ Efficient: do not search entire database to answer a query.
 - ▶ Convenient: allow users to create and query the data as easily as possible.
- ▶ **Secure, concurrent, and atomic** access
 - ▶ Allow multiple users to access database simultaneously.
 - ▶ Allow a user access to only to authorised data.
 - ▶ Provide some guarantee of reliability against system failures.

A Brief History of DBMS

- ▶ File systems

A Brief History of DBMS

- ▶ File systems
- ▶ The earliest databases (1960s) evolved from file systems
 - ▶ Navigational and hierarchical
 - ▶ User programmed the queries by walking from node to node in the DBMS.

A Brief History of DBMS

- ▶ File systems
- ▶ The earliest databases (1960s) evolved from file systems
 - ▶ Navigational and hierarchical
 - ▶ User programmed the queries by walking from node to node in the DBMS.
- ▶ Relational DBMS (1970s to now)
 - ▶ View database in terms of relations or tables
 - ▶ High-level query and definition languages such as SQL
 - ▶ Allow user to specify what she wants, not how to get what she wants

A Brief History of DBMS

- ▶ File systems
- ▶ The earliest databases (1960s) evolved from file systems
 - ▶ Navigational and hierarchical
 - ▶ User programmed the queries by walking from node to node in the DBMS.
- ▶ Relational DBMS (1970s to now)
 - ▶ View database in terms of relations or tables
 - ▶ High-level query and definition languages such as SQL
 - ▶ Allow user to specify what she wants, not how to get what she wants
- ▶ Object-oriented DBMS (1980s)
 - ▶ inspired by object-oriented languages
 - ▶ object-relational DBMs

A Brief History of DBMS

- ▶ File systems
- ▶ The earliest databases (1960s) evolved from file systems
 - ▶ Navigational and hierarchical
 - ▶ User programmed the queries by walking from node to node in the DBMS.
- ▶ Relational DBMS (1970s to now)
 - ▶ View database in terms of relations or tables
 - ▶ High-level query and definition languages such as SQL
 - ▶ Allow user to specify what she wants, not how to get what she wants
- ▶ Object-oriented DBMS (1980s)
 - ▶ inspired by object-oriented languages
 - ▶ object-relational DBMs
- ▶ New types of data:
 - ▶ Semi-structured data (XML)
 - ▶ Images, sounds, videos
 - ▶ Data streams

The DBMS Industry

- ▶ A DBMS is a software system.
- ▶ Major DBMS vendors: IBM, Microsoft, Oracle, Sybase
- ▶ Free/Open-source DBMS: MySQL, PostgreSQL, Firebird.
 - ▶ Used by companies such as Google, Yahoo, Lycos, BASF.
- ▶ All are “relational” (or “object-relational”) DBMS.

Example Scenario

- ▶ RDBMS \equiv “Relational” DBMS
- ▶ The relational model uses relations or tables to structure data

Example Scenario

- ▶ RDBMS \equiv “Relational” DBMS
- ▶ The relational model uses relations or tables to structure data
- ▶ `ClassList` relation:

<i>Student</i>	<i>Course</i>	<i>Grade</i>
Hermione Grainger	Potions	A-
Draco Malfoy	Potions	B
Harry Potter	Potions	A
Ron Weasley	Potions	C

- ▶ Relation separates the logical view (externals) from the physical view (internals)

Example Scenario

- ▶ RDBMS \equiv “Relational” DBMS
- ▶ The relational model uses relations or tables to structure data
- ▶ `ClassList` relation:

<i>Student</i>	<i>Course</i>	<i>Grade</i>
Hermione Grainger	Potions	A-
Draco Malfoy	Potions	B
Harry Potter	Potions	A
Ron Weasley	Potions	C

- ▶ Relation separates the logical view (externals) from the physical view (internals)
- ▶ Simple query languages (SQL) for accessing/modifying data
 - ▶ Find all students whose grades are better than B.
 - ▶ `SELECT Student FROM ClassList WHERE Grade > “B”`

DBMS Architecture

