CSEN 501 – CSEN501 - Databases I

Lecture 1:
Introduction and Basic Concepts

Prof. Dr. Slim Abdennadher
Dr. Nada Sharaf

German University Cairo, Faculty of Media Engineering and Technology

Course Structure

1. Lectures
2. Exercises and Homework
   - Practical Assignments
   - Work in teams, use feedback from tutors
3. Labs
   - Supervised lab Assignments
   - Work in teams

WWW-page: Useful info and important announcements
met.guc.edu.eg

Tentative Grading

Overall weighting for your grade

- 5% for assignments
- 10% for quizzes
- 30% for project
- 25% for mid-term exam
- 30% for Final exam

What is a Database System (DBS)?

- A **Database (DB)** is a very large, integrated collection of data designed to meet the information needs of an organization.
- A DB models a real world enterprise.
  - Entities: e.g. students, courses . . .
  - Relationships: e.g. Dina is taking CSEN501
- A Database Management System (DBMS) is a software package designed to store and manage databases.
  - Allows the user to define the database through a Data Definition Language (DDL).
  - Allows users to insert, update, delete, and retrieve data from the database through a Data Manipulation Language (DML).
  - Provides controlled access to the database.
- Database System = DBMS + Data

Why Study Databases?

Canada Jobs, Government Jobs, Vacancy, Career 2010/2011

Ottawa Job Vacancy of Computer Science, Electrical Engineering in Hydro Ottawa

Written on August 9, 2010 – 8:17 am | by canadajobs

HYDRO OTTAWA.
Ottawa, ON, CANADA

Hydro Ottawa Limited, a wholly owned subsidiary of Hydro Ottawa Holding Inc. is the third largest municipal electricity distribution company in the province. Hydro Ottawa is responsible for the safe, reliable delivery of electricity to more than 291,000 residential and business customers in the city of Ottawa and the village of Casselman.

Meter Data Systems Senior Analyst

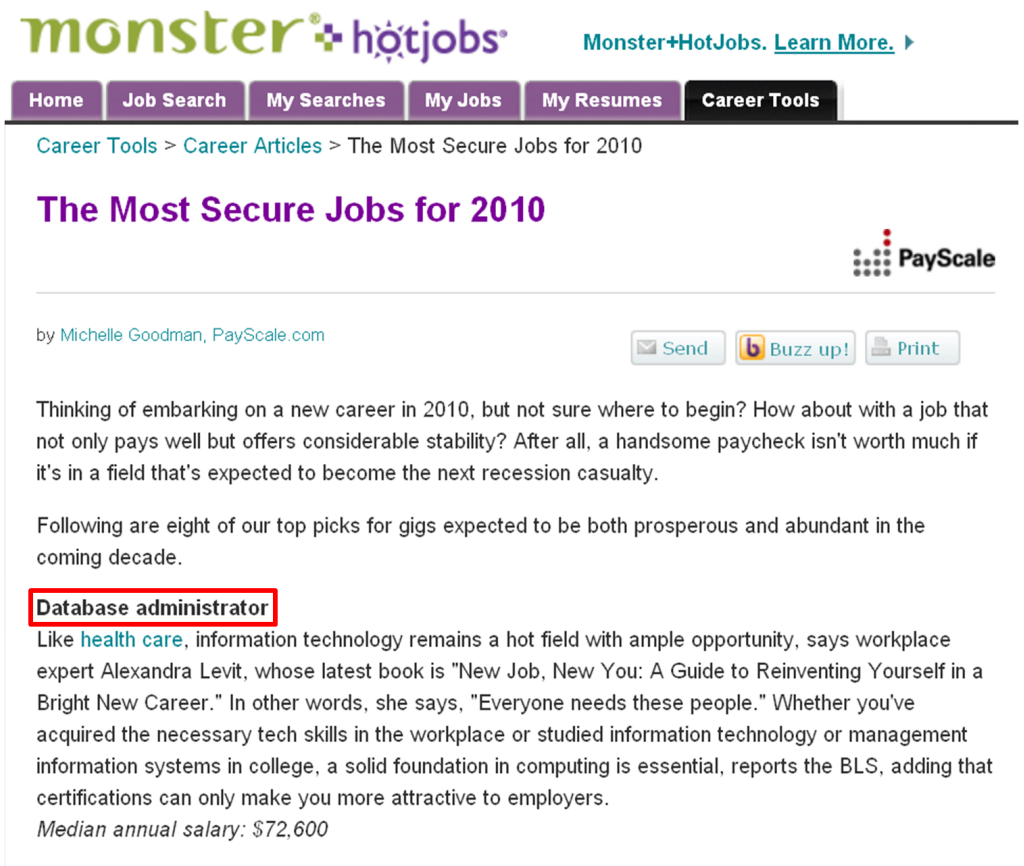
Education/Experience:

Post Secondary Graduate of Computer Science or Electrical Engineering or related discipline.

Proven knowledge of computer software applications and database tools (MS-Access, Oracle, SQL, Visual Basic) with a superior ability to create, maintain and link database applications.

Thorough working knowledge of Microsoft Office Suite Products.

Why Study Databases?



monster hotjobs Monster+HotJobs. Learn More.

Home | Job Search | My Searches | My Jobs | My Resumes | Career Tools

Career Tools > Career Articles > The Most Secure Jobs for 2010

The Most Secure Jobs for 2010

PayScale

by Michelle Goodman, PayScale.com

Send | Buzz up! | Print

Thinking of embarking on a new career in 2010, but not sure where to begin? How about with a job that not only pays well but offers considerable stability? After all, a handsome paycheck isn't worth much if it's in a field that's expected to become the next recession casualty.

Following are eight of our top picks for gigs expected to be both prosperous and abundant in the coming decade.

Database administrator

Like health care, information technology remains a hot field with ample opportunity, says workplace expert Alexandra Levit, whose latest book is "New Job, New You: A Guide to Reinventing Yourself in a Bright New Career." In other words, she says, "Everyone needs these people." Whether you've acquired the necessary tech skills in the workplace or studied information technology or management information systems in college, a solid foundation in computing is essential, reports the BLS, adding that certifications can only make you more attractive to employers.

Median annual salary: $72,600

Good Jobs

Database Administrator
Median Salary: **$87,330**
No. of Active Positions on Switch: **243**

Database administrators are the highest-paid security guards on the market today, and we mean that in the most flattering sense possible. These tech junkies are responsible for the security and monitoring of databases. They also handle configuration and administration of these databases, which are essentially the Rosetta Stone for tech companies -- particularly startups.

The Most Controversial Buzzword

“Knowledge is of two kinds: we know a subject ourselves, or we know where we can find information upon it.”’ Samuel Johnson (1709-1784)

Database Systems

- Information is one of the most valuable resources in this information age!
- How do we effectively and efficiently manage this information?
  - How does Wal-Mart manage its 200 TB data warehouse?
  - What is the database technology behind ebay's website?
  - How do you build an Oracle 10g, IBM DB2 or Microsoft SQL Server database?
- Relational Database Management Systems: Dominant data management paradigm today
- 10 Billion dollar a year industry (in the U.S. only): You will see this in the job market!

Why Databases

- Shift from computation to information
- Need for DBMS has exploded: corporate and scientific.
- Datasets increasing in diversity and volume such as: Genome project, Earth Observation System project

Typical Applications of DB: Amazon

What the subject is about

This course covers the fundamental concepts in database systems:

- Organization of data
- Efficient retrieval of data
- Reliable storage of data
- Maintaining consistent data
- Not surprisingly, all these topics are interrelated.

Why Study Databases?

- History of database research over the past 40 years led to the DBS becoming the most important development in the field of software engineering.
- The apparent simplicity of these systems has led to users creating ineffective and inefficient database systems and applications.

SOFTWARE CRISIS
SOFTWARE DEPRESSION

Course Goal: Be Ready to Lead

- Understand the foundations on which today's work is built
  - Existing database systems
  - Research
- Know enough to
  - Participate in building the next DBMS
  - Be prepared to develop the theory behind the one after that

Why don't we "program up"' databases when we need them?

- For simple and small databases this is often the best solution: Flat files and grep get us a long way.
- We run into problems when
  - The structure is complicated (more than a simple table)
  - The database gets large
  - Many people want to use it simultaneously

Example: Personal Calendar

- We might start by building a file with the following structure:

What	Day	When	Who	Where
Lunch	9/17	1pm	XY	Mariott
CSEN501	9/18	11 am	Dr. Slim	H5
Biking	9/19	9 am.	Amir	Amir's house
Dinner	9/19	6pm	Amir	Cafe Paris

- This text file is easy to deal with. So there is no need for a DBMS!

Problem 1: Data Organization

- Consider the important “who” field. Do we also want to keep e-mailaddresses, telephone numbers, . . . ?
- Expand the file to look like:
  What Day When Who-name Who-email Who-tel . . .
- Now we are keeping our address book in our calendar and doing so redundantly.

“Link” Calendar with Address Book?

- Two conceptual entities - contact information and calendar - with a relationship between them, linking people in the calendar to their contact information.
- This link could be based on something as simple as the person’s name.

Problem 2: Efficiency

- Size of personal address book is probably less than one hundred entries, but there are things we would like to do quickly and efficiently.
  - “Give me all appointments on 9/18”
  - “When am I next meeting Amir?”
- “Program”’ these as quickly as possible.
- Have these programs executed efficiently.
- What would happen if you were using a “corporate” calendar with hundreds of thousands of entries?

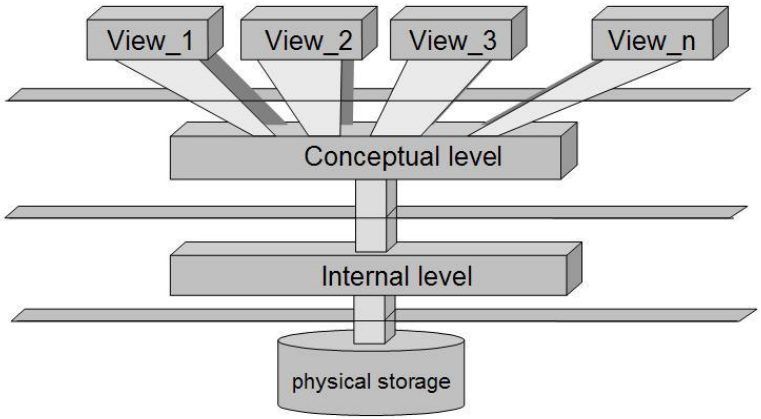
Problem 3: Concurrency and Reliability

- **Concurrency**: Suppose other people are allowed to access your calendar and are allowed to modify it? How do we stop two people changing the file at the same time and leaving it in a physical (or logical) mess?
- Example: Suppose I schedule a meeting with a student after class today (3:00pm) and at the same time my secretary schedules me to meet with the Chairman. We both see that the time is open, but presumably only one of the two meetings will show on the calendar later.
- **Reliability**: Suppose the system crashes while we are changing the calendar. How do we recover our work?

Transactions and Log

- Key concept for concurrency is that of a transaction : an atomic sequence of database actions (read/write) on data items (e.g. calendar entry).
- Key concept for recoverability is that of a log: keeping track of all actions carried out by the DB.

Database architecture - the traditional view



It is common to describe databases in two ways:

- The **logical level**: What users see, i.e. the program or query language interface.
- It is traditional to split the logical level into two components:
  - **Conceptual level**: overall database design
  - **Views that various users get to see.**
- The **physical level**: How files are organized, what indexing mechanisms are used.

Data independence

- Data independence: one of the most important benefits of DBMS
- A user of a relational database system should be able to use SQL to query the database without knowing about how precisely data is stored, e.g.

$$\begin{array}{c} \textit{SELECT When, Where} \\ \textit{FROM Calendar} \\ \textit{WHERE Who} = \textit{'' Bill''} \end{array}$$

- After all, you don't worry much how numbers are used in a computer-based calculator.
- We distinguish between two kinds of data independence:
  - Logical data independence - protects the user from changes in the logical structure of the data - could completely reorganize the calendar "schema" without changing how I query it.
  - Physical data independence protects the user from changes in the physical structure of data: could add an index on **Who** without changing how the user would write the query, but the query would execute faster (query optimization).

Roles in the Database Environment

- Data and Database Administrators / Database Designers
  - Designs the logical/physical schemas
  - Handles Security and authorization
  - Crash Recovery
  - Database tuning as needs evolve
- Application Programmers: Provide the required functionality to the end-users
- End-Users: Clients for the database
  - Naive users: unaware of the DBMS
  - Sophisticated users: Familiar with the structure of the database and the facilities offered by the DBMS.

History: Traditional File-Based Systems

- File-Based System: Predecessor of Database system
- File-Based System: A collection of application programs that performs services for the end-users.
- Each program defines and manages its own data.
- Limitations of the File-Based Approach
  - Separation and isolation of Data
  - Duplication of Data
  - Data Dependence
  - Incompatibility offiles
  - Fixed queries

Is the WWW a DBMS?

- Sophisticated search available
  - Crawler indexes pages on the web
  - Availability of Keyword-based search for pages

  But...
- Available data is most of the times unstructured and untyped
  - Can only search (can't modify, summarize, analyze, correlate, . . .)
  - No guarantees of freshness, accuracy, durability, consistency
  - DBMS behind most Web sites provide such functions
- The picture is changing
  - New standards like XML can help data modeling
  - The WWW/DB boundary is blurry!

Is file system a DBMS?

- Strong shared heritage
  - Descendant of file management system
  - Excellent insulator against hardware changes
- However. . .
- Data mostly unstructured and untyped
- No constraints or relationships
- Minimal support for isolation, consistency
- The picture is changing
  - File systems are adopting database concepts such as logging, transactions
  - Object-oriented file systems provide users with finer grain data models
  - The FS/DBMS boundary is blurry!

Databases help:

- DBMS vendors and programmers: Oracle, IBM, MS, Sybase, NCR, . . .
- End users: in fields like education, science and in business . . .
- DB application programmers
  - They can build applications on top of DBMSs
  - They can build web services that run through DBMSs
- Database administrators (DBAs)
  - Design logical and physical schemas
  - Handle security and authorization issues
  - Handle data availability, crash recovery
  - Tune the Database as needs evolve

Historical Development of Database Technology

- Early Database Applications: The hierarchical and Network models (mid 1960's and dominated the seventies)
- Relational Model Based Systems: The model was introduced 1970 by E.F. Codd (IBM)
- Relational DBMS products emerged in the 1980's
- Object Oriented DBMS: late 1980's to cater to the need of complex data processing in CAD and other applications.
- Data on the web and E-Commerce applications: XML Database
- Future trends: Huge (terabyte) systems are appearing and will require novel means of handling and analyzing data, e.g. genome project, geological, national security, and space exploration data.

Advantages and Disadvantages of DBMS

- Advantages:
  - Control of Data Redundancy
  - Data Consistency
  - Sharing of Data
  - Improved Data Integrity and Security
  - Increased Concurrency
- Disadvantages:
  - Complexity
  - Size
  - Cost of DBMS
  - Additional Hardware Costs
  - Performance
  - Higher Impact of Failure