


Database Systems

Session 1 – Main Theme
Introduction to Database Systems
Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Presentation material partially based on textbook slides
Fundamentals of Database Systems (6th Edition)
by Ramez Elmasri and Shamkant Navathe
Slides copyright © 2011

Agenda

- 
- 1 Instructor and Course Introduction
 - 2 Introduction to Database Systems
 - 3 Database System Architecture
 - 4 Summary and Conclusion

Who am I?



- Profile -

- 27 years of experience in the Information Technology Industry, including twelve years of experience working for leading IT consulting firms such as Computer Sciences Corporation
- PhD in Computer Science from University of Colorado at Boulder
- Past CEO and CTO
- Held senior management and technical leadership roles in many large IT Strategy and Modernization projects for fortune 500 corporations in the insurance, banking, investment banking, pharmaceutical, retail, and information management industries
- Contributed to several high-profile ARPA and NSF research projects
- Played an active role as a member of the OMG, ODMG, and X3H2 standards committees and as a Professor of Computer Science at Columbia initially and New York University since 1997
- Proven record of delivering business solutions on time and on budget
- Original designer and developer of jcrew.com and the suite of products now known as IBM InfoSphere DataStage
- Creator of the Enterprise Architecture Management Framework (EAMF) and main contributor to the creation of various maturity assessment methodology
- Developed partnerships between several companies and New York University to incubate new methodologies (e.g., EA maturity assessment methodology developed in Fall 2008), develop proof of concept software, recruit skilled graduates, and increase the companies' visibility

3

How to reach me?



	Cell	(212) 203-5004
	Email	jcf@cs.nyu.edu
	AIM, Y! IM, ICQ	jcf2_2003
	MSN IM	jcf2_2003@yahoo.com
	LinkedIn	http://www.linkedin.com/in/jcfranchitti
	Twitter	http://twitter.com/jcfranchitti
	Skype	jcf2_2003@yahoo.com

4

What is the class about?



- Course description and syllabus:

- » <http://www.nyu.edu/classes/jcf/CSCI-GA.2433-001>
- » <http://cs.nyu.edu/courses/fall11/CSCI-GA.2433-001/>

- Textbooks:

- » ***Fundamentals of Database Systems (6th Edition)***

Ramez Elmasri and Shamkant Navathe

Addison Wesley

ISBN-10: 0-1360-8620-9, ISBN-13: 978-0136086208 6th Edition (04/10)



5

Icons / Metaphors



Information



Common Realization



Knowledge/Competency Pattern



Governance



Alignment



Solution Approach

6

Course Objectives



- Gain understanding of fundamental concepts of state-of-the art databases (more precisely called: Database Management Systems)
- Get to know some of the tools used in the design and implementations of databases
- Know enough so that it is possible to read/skim a database system manual and
 - Start designing and implementing small data bases
 - Start managing and interacting with existing small to large databases
- Experiment and practice with industry leading vendor solutions:
 - CA's Erwin for design of relational database
 - Oracle, IBM DB2, and other DB products for writing relational queries

7

Key Material Covered (1/2)



- Methodology used for modeling a business application during the database design process, focusing on entity-relationship model and entity relationship diagrams
- Relational model and implementing an entity-relationship diagram
- Relational algebra (using SQL syntax)
- SQL as data manipulation language
- SQL as data definition language
- Refining a relational implementation, including the normalization process and the algorithms to achieve normalization

8

Key Material Covered (2/2)



- Physical design of the database using various file organization and indexing techniques for efficient query processing
- Concurrency Control
- Recovery
- Query execution
- Data warehouses
- Additional topics may be covered as time allows, these topics are covered in greater depth in other courses but PowerPoint presentations for them will still be provided
- The course material is partially derived from the textbook slides and material covered as part of the Database Systems course offered at NYU Courant in previous semesters

9


Software Requirements



- Microsoft Windows XP (Professional Ed.) / Vista / 7
- Software tools will be available from the Internet or from the course Web site under demos as a choice of freeware or commercial tools
 - Database Modeling Tools
 - Database Management Software Tools
 - etc.
- References will be provided on the course Web site

10

Agenda

- 
- 1 Instructor and Course Introduction
 - 2 Introduction to Database Systems
 - 3 Database System Architecture
 - 4 Summary and Conclusion

11

Section Outline



- Introduction
- An Example
- Characteristics of the Database Approach
- Actors on the Scene
- Workers behind the Scene
- Advantages of Using the DBMS Approach
- A Brief History of Database Applications
- When Not to Use a DBMS

12



- Traditional database applications
 - Store textual or numeric information
- Multimedia databases
 - Store images, audio clips, and video streams digitally
- Geographic information systems (GIS)
 - Store and analyze maps, weather data, and satellite images
- Data warehouses and online analytical processing (OLAP) systems
 - Extract and analyze useful business information from very large databases
 - Support decision making
- Real-time and active database technology
 - Control industrial and manufacturing processes



- Database
 - Collection of related data
 - Known facts that can be recorded and that have implicit meaning
 - Mini-world or Universe of Discourse (UoD)
 - Represents some aspect of the real world
 - Logically coherent collection of data with inherent meaning
 - Built for a specific purpose
- Example of a large commercial database
 - Amazon.com
- Database management system (DBMS)
 - Collection of programs
 - Enables users to create and maintain a database
- Defining a database
 - Specify the data types, structures, and constraints of the data to be stored



- **Meta-data**
 - Database definition or descriptive information
 - Stored by the DBMS in the form of a database catalog or dictionary
- **Manipulating a database**
 - Query and update the database miniworld
 - Generate reports
- **Sharing a database**
 - Allow multiple users and programs to access the database simultaneously
- **Application program**
 - Accesses database by sending queries to DBMS
- **Query**
 - Causes some data to be retrieved



- **Transaction**
 - May cause some data to be read and some data to be written into the database
- **Protection includes:**
 - System protection
 - Security protection
- **Maintain the database system**
 - Allow the system to evolve as requirements change over time

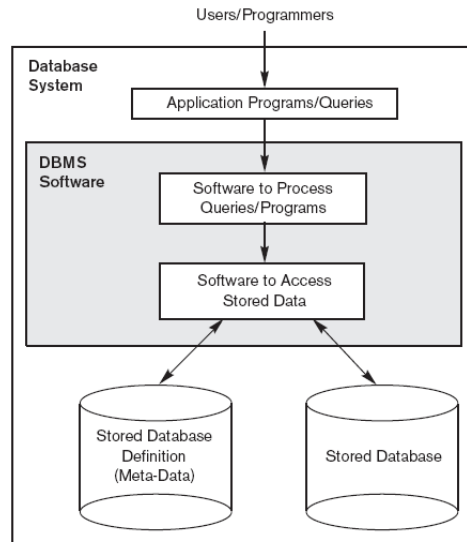


Figure 1.1
A simplified database system environment.



- **UNIVERSITY database**
 - Information concerning students, courses, and grades in a university environment
- **Data records**
 - STUDENT
 - COURSE
 - SECTION
 - GRADE_REPORT
 - PREREQUISITE
- Specify structure of records of each file by specifying data type for each data element
 - String of alphabetic characters
 - Integer
 - etc.

High-Level Example – Database Implementation (1/2)



- Construct UNIVERSITY database
 - Store data to represent each student, course, section, grade report, and prerequisite as a record in appropriate file
- Relationships among the records
- Manipulation involves querying and updating
- Examples of queries:
 - Retrieve the transcript
 - List the names of students who took the section of the 'Database' course offered in fall 2008 and their grades in that section
 - List the prerequisites of the 'Database' course
- Examples of updates:
 - Change the class of 'Smith' to sophomore
 - Create a new section for the 'Database' course for this semester
 - Enter a grade of 'A' for 'Smith' in the 'Database' section of last semester
- Phases for designing a database:
 - Requirements specification and analysis
 - Conceptual design
 - Logical design
 - Physical design

19

High-Level Example – Database Implementation (2/2)



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

20

Characteristics of the Database Approach



- Traditional file processing
 - Each user defines and implements the files needed for a specific software application
- Database approach
 - Single repository maintains data that is defined once and then accessed by various users
- Main characteristics of database approach
 - Self-describing nature of a database system
 - Insulation between programs and data, and data abstraction
 - Support of multiple views of the data
 - Sharing of data and multiuser transaction processing

21

Self-Describing Nature of a Database System



- Database system contains complete definition of structure and constraints
- Meta-data
 - Describes structure of the database
- Database catalog used by:
 - DBMS software
 - Database users who need information about database structure

RELATIONS

Relation_name	No. of columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXXXXX	COURSE
....
....
....
....
Prerequisite_number	XXXXXXXX	PREREQUISITE

Note: Major_type is defined as an enumerated type with all known majors. XXXXXXXX is used to define a type with four alpha characters followed by four digits.

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

22



- Program-data independence
 - Structure of data files is stored in DBMS catalog separately from access programs
- Program-operation independence
 - Operations specified in two parts:
 - Interface includes operation name and data types of its arguments
 - Implementation can be changed without affecting the interface



- Data abstraction
 - Allows program-data independence and program-operation independence
- Conceptual representation of data
 - Does not include details of how data is stored or how operations are implemented
- Data model
 - Type of data abstraction used to provide conceptual representation



- View
 - Subset of the database
 - Contains virtual data derived from the database files but is not explicitly stored
- Multiuser DBMS
 - Users have a variety of distinct applications
 - Must provide facilities for defining multiple views



- Allow multiple users to access the database at the same time
- Concurrency control software
 - Ensure that several users trying to update the same data do so in a controlled manner
 - Result of the updates is correct
- Online transaction processing (OLTP) application
- Transaction
 - Central to many database applications
 - Executing program or process that includes one or more database
 - Isolation property
 - Each transaction appears to execute in isolation from other transactions
 - Atomicity property
 - Either all the database operations in a transaction are executed or none are

Actors on the Scene



- Database administrators (DBA) are responsible for:
 - Authorizing access to the database
 - Coordinating and monitoring its use
 - Acquiring software and hardware resources
- Database designers are responsible for:
 - Identifying the data to be stored
 - Choosing appropriate structures to represent and store this data
- End users
 - People whose jobs require access to the database
 - Types
 - Casual end users
 - Naive or parametric end users
 - Sophisticated end users
 - Standalone users
- System analysts
 - Determine requirements of end users
- Application programmers
 - Implement these specifications as programs

27

Workers behind the Scene



- DBMS system designers and implementers
 - Design and implement the DBMS modules and interfaces as a software package
- Tool developers
 - Design and implement tools
- Operators and maintenance personnel
 - Responsible for running and maintenance of hardware and software environment for database system

28

Advantages of Using the DBMS Approach (1/3)



- Controlling redundancy
 - Data normalization
 - Denormalization
 - Sometimes necessary to use controlled redundancy to improve the performance of queries
- Restricting unauthorized access
 - Security and authorization subsystem
 - Privileged software
- Providing persistent storage for program objects
 - Complex object in C++ can be stored permanently in an object-oriented DBMS
 - Impedance mismatch problem
 - Object-oriented database systems typically offer data structure compatibility
- Providing storage structures and search techniques for efficient query processing
 - Indexes
 - Buffering and caching
 - Query processing and optimization

29

Advantages of Using the DBMS Approach (2/3)



- Providing backup and recovery
 - Backup and recovery subsystem of the DBMS is responsible for recovery
- Providing multiple user interfaces
 - Graphical user interfaces (GUIs)
- Representing complex relationships among data
 - May include numerous varieties of data that are interrelated in many ways
- Enforcing integrity constraints
 - Referential integrity constraint
 - Every section record must be related to a course record
 - Key or uniqueness constraint
 - Every course record must have a unique value for Course_number
 - Business rules
 - Inherent rules of the data model

30



- Permitting inferencing and actions using rules
 - Deductive database systems
 - Provide capabilities for defining deduction rules
 - Inferencing new information from the stored database facts
 - Trigger
 - Rule activated by updates to the table
 - Stored procedures
 - More involved procedures to enforce rules
- Additional implications of using the database approach
 - Reduced application development time
 - Flexibility
 - Availability of up-to-date information
 - Economies of scale



- Early database applications using hierarchical and network systems
 - Large numbers of records of similar structure
- Providing data abstraction and application flexibility with relational databases
 - Separates physical storage of data from its conceptual representation
 - Provides a mathematical foundation for data representation and querying
- Object-oriented applications and the need for more complex databases
 - Used in specialized applications: engineering design, multimedia publishing, and manufacturing systems



- Interchanging data on the Web for e-commerce using XML
 - Extended markup language (XML) primary standard for interchanging data among various types of databases and Web pages
- Extending database capabilities for new applications
 - Extensions to better support specialized requirements for applications
 - Enterprise resource planning (ERP)
 - Customer relationship management (CRM)
- Databases versus information retrieval
 - Information retrieval (IR)
 - Deals with books, manuscripts, and various forms of library-based articles



- More desirable to use regular files for:
 - Simple, well-defined database applications not expected to change at all
 - Stringent, real-time requirements that may not be met because of DBMS overhead
 - Embedded systems with limited storage capacity
 - No multiple-user access to data

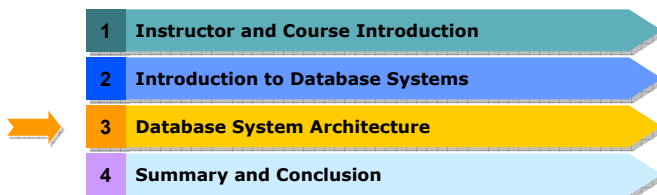
Summary



- Database
 - Collection of related data (recorded facts)
- DBMS
 - Generalized software package for implementing and maintaining a computerized database
- Several categories of database users
- Database applications have evolved
 - Current trends: IR, Web

35

Agenda



36



- Data Models, Schemas, and Instances
- Three-Schema Architecture and Data Independence
- Database Languages and Interfaces
- The Database System Environment
- Centralized and Client/Server Architectures for DBMSs
- Classification of Database Management Systems



- Basic client/server DBMS architecture
 - Client module
 - Server module



- Data abstraction
 - Suppression of details of data organization and storage
 - Highlighting of the essential features for an improved understanding of data
- Data model
 - Collection of concepts that describe the structure of a database
 - Provides means to achieve data abstraction
 - Basic operations
 - Specify retrievals and updates on the database
 - Dynamic aspect or behavior of a database application
 - Allows the database designer to specify a set of valid operations allowed on database objects



- High-level or conceptual data models
 - Close to the way many users perceive data
- Low-level or physical data models
 - Describe the details of how data is stored on computer storage media
- Representational data models
 - Easily understood by end users
 - Also similar to how data organized in computer storage
- Entity
 - Represents a real-world object or concept
- Attribute
 - Represents some property of interest
 - Further describes an entity
- Relationship among two or more entities
 - Represents an association among the entities
 - Entity-Relationship model

Categories of Data Models (2/2)



- Relational data model
 - Used most frequently in traditional commercial DBMSs
- Object data model
 - New family of higher-level implementation data models
 - Closer to conceptual data models
- Physical data models
 - Describe how data is stored as files in the computer
 - Access path
 - Structure that makes the search for particular database records efficient
 - Index
 - Example of an access path
 - Allows direct access to data using an index term or a keyword

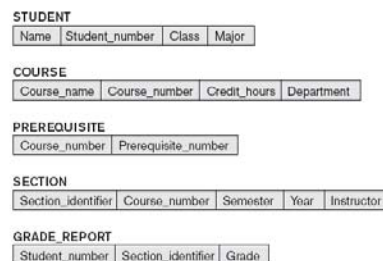
41

Schemas, Instances, and Database State (1/2)



- Database schema
 - Description of a database
- Schema diagram
 - Displays selected aspects of schema
- Schema construct
 - Each object in the schema
- Database state or snapshot
 - Data in database at a particular moment in time

Figure 2.1
Schema diagram for the database in Figure 1.2.



⁶Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

⁷It is customary in database parlance to use schemas as the plural for schema, even though schemata is the proper plural form. The word *scheme* is also sometimes used to refer to a schema.

42

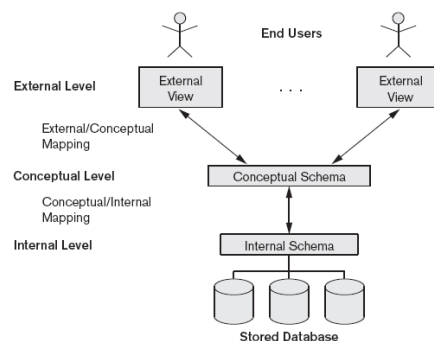


- Define a new database
 - Specify database schema to the DBMS
- Initial state
 - Populated or loaded with the initial data
- Valid state
 - Satisfies the structure and constraints specified in the schema
- Schema evolution
 - Changes applied to schema as application requirements change



- Internal level
 - Describes physical storage structure of the database
- Conceptual level
 - Describes structure of the whole database for a community of users
- External or view level
 - Describes part of the database that a particular user group is interested in

Figure 2.2
The three-schema architecture.





- Capacity to change the schema at one level of a database system
 - Without having to change the schema at the next higher level
- Types:
 - Logical
 - Physical



- Data definition language (DDL)
 - Defines both schemas
- Storage definition language (SDL)
 - Specifies the internal schema
- View definition language (VDL)
 - Specifies user views/mappings to conceptual schema
- Data manipulation language (DML)
 - Allows retrieval, insertion, deletion, modification
- High-level or nonprocedural DML
 - Can be used on its own to specify complex database operations concisely
 - Set-at-a-time or set-oriented
- Low-level or procedural DML
 - Must be embedded in a general-purpose programming language
 - Record-at-a-time



- Menu-based interfaces for Web clients or browsing
- Forms-based interfaces
- Graphical user interfaces
- Natural language interfaces
- Speech input and output
- Interfaces for parametric users
- Interfaces for the DBA



- DBMS component modules
 - Buffer management
 - Stored data manager
 - DDL compiler
 - Interactive query interface
 - Query compiler
 - Query optimizer
 - Pre-compiler
 - Runtime database processor
 - System catalog
 - Concurrency control system
 - Backup and recovery system

Component Modules of a DBMS

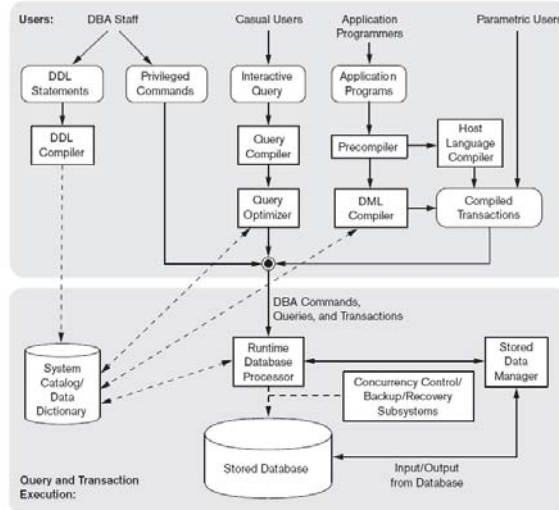


Figure 2.3
Component modules of a DBMS and their interactions.

Database System Utilities



- **Loading**
 - Load existing data files
- **Backup**
 - Creates a backup copy of the database
- **Database storage reorganization**
 - Reorganize a set of database files into different file organizations
- **Performance monitoring**
 - Monitors database usage and provides statistics to the DBA



- CASE Tools
- Data dictionary (data repository) system
 - Stores design decisions, usage standards, application program descriptions, and user information
- Application development environments
- Communications software



- Centralized DBMSs Architecture
 - All DBMS functionality, application program execution, and user interface processing carried out on one machine

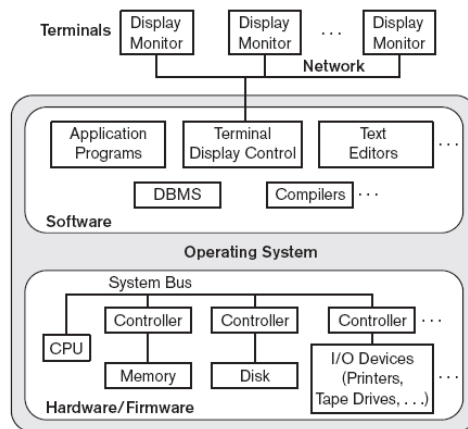


Figure 2.4
A physical centralized architecture.



- Servers with specific functionalities
 - File server
 - Maintains the files of the client machines.
 - Printer server
 - Connected to various printers; all print requests by the clients are forwarded to this machine
 - Web servers or e-mail servers
- Client machines
 - Provide user with:
 - Appropriate interfaces to utilize these servers
 - Local processing power to run local applications



- Client
 - User machine that provides user interface capabilities and local processing
- Server
 - System containing both hardware and software
 - Provides services to the client machines
 - Such as file access, printing, archiving, or database access

Sample Two-Tier Client / Server Architecture

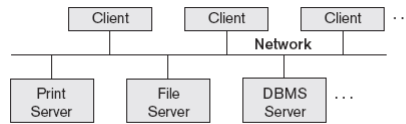


Figure 2.5
Logical two-tier
client/server
architecture.

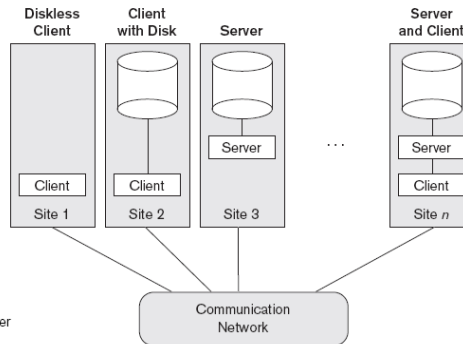


Figure 2.6
Physical two-tier client/server
architecture.

55

Two-Tier Client/Server Architectures for DBMSs



- **Server handles**
 - Query and transaction functionality related to SQL processing
- **Client handles**
 - User interface programs and application programs
- **Open Database Connectivity (ODBC)**
 - Provides application programming interface (API)
 - Allows client-side programs to call the DBMS
 - Both client and server machines must have the necessary software installed
- **JDBC**
 - Allows Java client programs to access one or more DBMSs through a standard interface

56



- Application server or Web server
 - Adds intermediate layer between client and the database server
 - Runs application programs and stores business rules
- N-tier
 - Divide the layers between the user and the stored data further into finer components

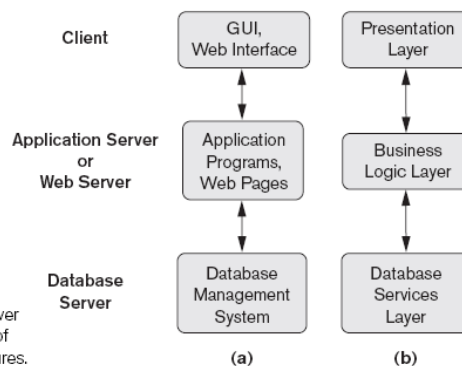


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

Classification of Database Management Systems



- Data model
 - Relational
 - Object
 - Hierarchical and network (legacy)
 - Native XML DBMS
- Number of users
 - Single-user
 - Multiuser
- Number of sites
 - Centralized
 - Distributed
 - Homogeneous
 - Heterogeneous
- Cost
 - Open source
 - Different types of licensing

59

Classification of Database Management Systems



- Types of access path options
- General or special-purpose

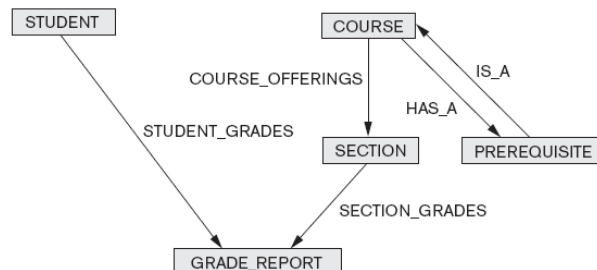


Figure 2.8
The schema of Figure 2.1 in network model notation.

¹⁴CODASYL DBTG stands for Conference on Data Systems Languages Database Task Group, which is the committee that specified the network model and its language.

60

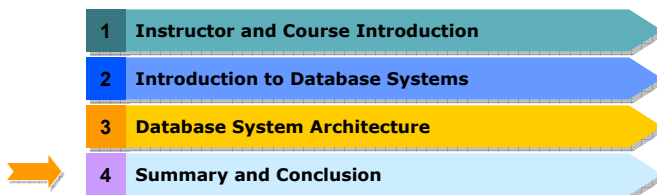
Section Summary



- Concepts used in database systems
- Main categories of data models
- Types of languages supported by DBMSs
- Interfaces provided by the DBMS
- DBMS classification criteria:
 - Data model, number of users, number of sites, access paths, cost

61

Agenda



62

Course Assignments



- Individual Assignments
 - Reports based on case studies / class presentations
 - Textbook problem sets
- Project-Related Assignments
 - All assignments (other than the individual assessments) will correspond to milestones in the course project

63

Assignments & Readings



- Readings
 - » Slides and Handouts posted on the course web site
 - » Textbook: Chapters 1 & 2

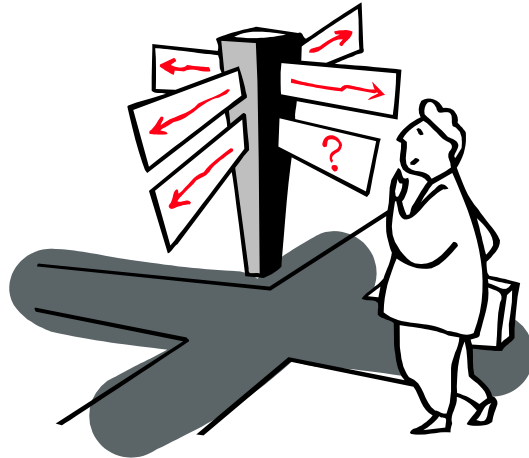


64

Next Session: Relational Data Model and Relational Database Constraints

65

Any Questions?



66