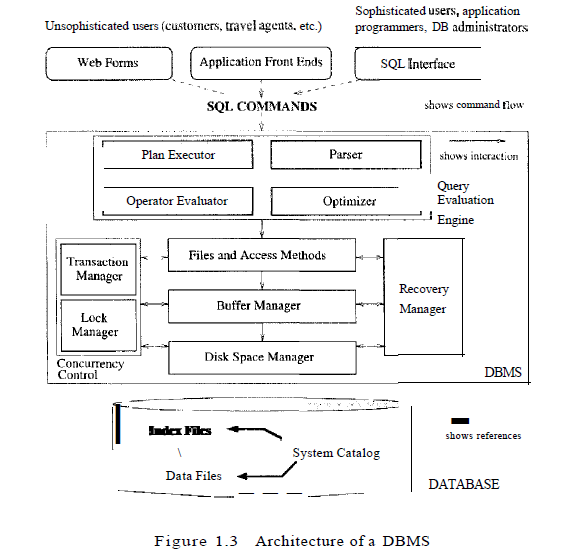
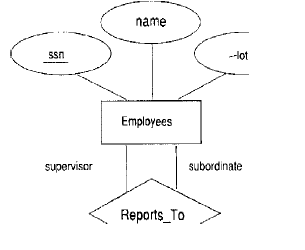
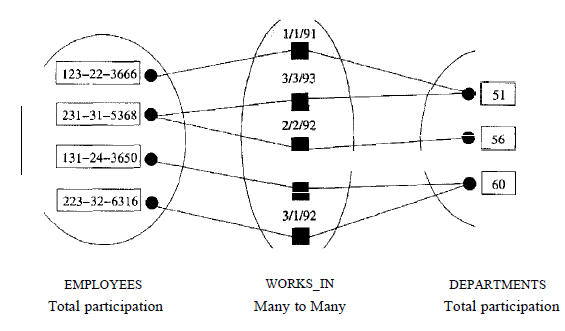
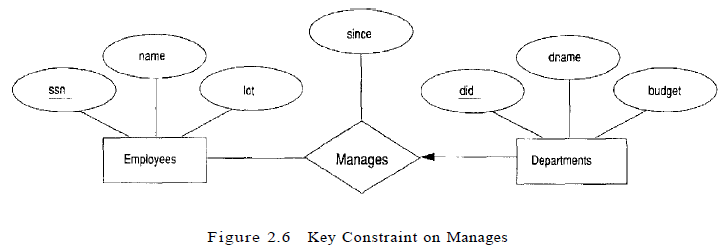
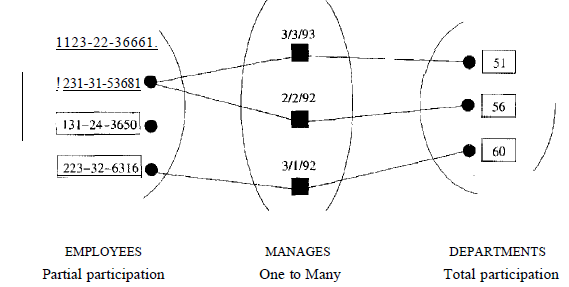
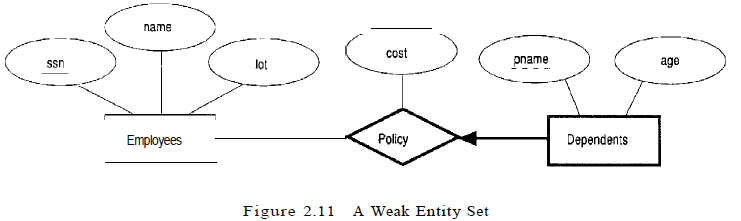
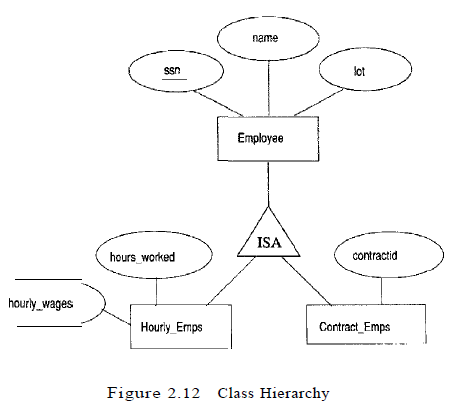
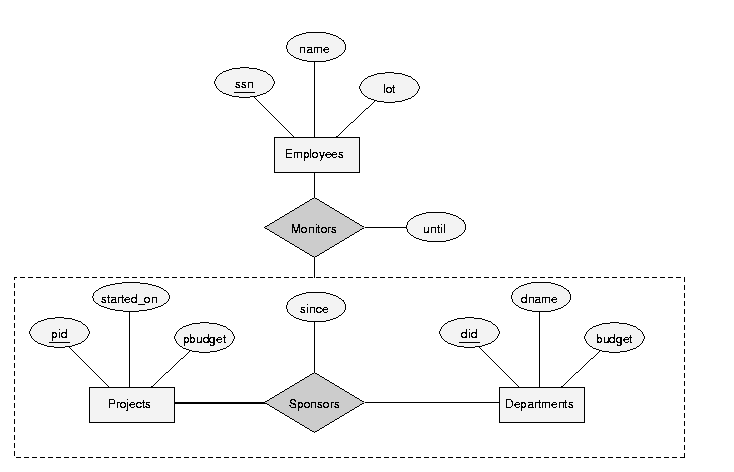
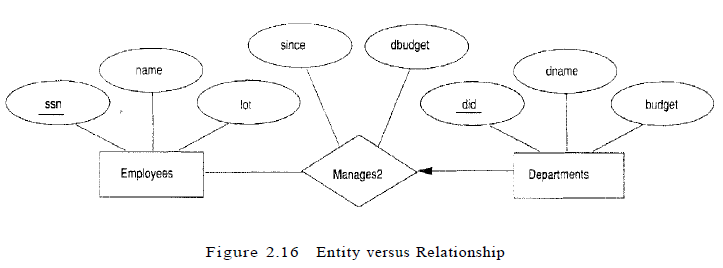
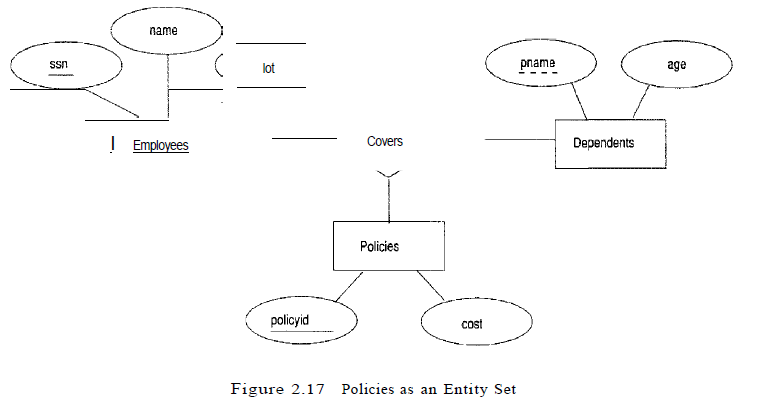
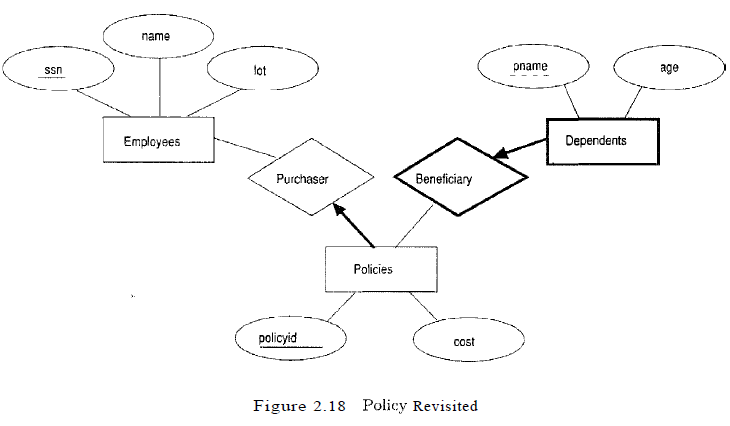
DBMS (Chapter 1)

* Database
  + A collection of data, typically describing the activities of one or more related organizations
* Database management system (DBMS)
  + Software designed to assist in maintaining and utilizing large collection of data
* Drawbacks of using operating system for managing data
  + Do not have 500GB of main memory to hold all the data
    - Even if it has 500 GB of main memory, on computer systems with 32-bit addressing, we cannot refer directly to more thana bout 4 GB of data
  + Have to write special programs to answer each question a user may want to ask about the data
  + Must protect the data from inconsistent changes made by different users accessing the data concurrently
  + Operating systems provide only a password mechanism for security
  + Must ensure that the data is restored to a consistent state if the system crashes while changes are being made
* Advantages of DBMS
  + Data independence
    - DBMS provides an abstract view of the data that hides such details
  + Efficient data access
    - DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently
  + Data integrity and security
    - DBMS can enforce integrity constraints
    - DBMS can enforce access controls that govern what data is visible to different classes of users
  + Data administration
    - Centralizing the administration of data can offer significant improvements
      * Minimize redundancy
      * Fine-tuning the storage to make retrieval efficient
  + Concurrent access and crash recovery
    - DBMS schedules concurrent accesses to the data in such a manner that users can think of data as being accessed by only one user at a time
    - DBMS protects users from the effects of system failures
  + Reduced application development time
    - DBMS handles many important tasks
      * Users do not need to debug or test in the application
* Limitations of DBMS
  + Complex piece of software
  + Optimized for certain kinds of workloads
  + Performance may not be adequate for certain specialized applications
  + Application may need to manipulate the data in ways not supported by the query language
* Data model
  + Definition
    - A collection of high-level data description constructs that hide many low-level storage details
  + Relational data model
    - DBMS is typically built around a data model with just a few basic constructs
  + Semantic data model
    - More abstract, high-level data model that makes it easier for a user to come up with a good initial description of the data in an enterprise
    - Serves as a useful starting point and is subsequently translated into a database design in terms of the data model the DBMS actually supports
  + Relational Model
    - Definitions
      * Relation
        + Set of records
      * Schema
        + Description of data in terms of a data model
      * Integrity constraints
        + Conditions that the records in a relation must satisfy
  + Other data models
    - Hierarchical model (Used in IBM’s IMS DBMS)
    - Network model (Used in IDS and IDMS)
    - Object-oriented model (Used in ObjectStore and Versant)
    - Object-relational model (Used in DBMS products from IBM, Informix, ObjectStore, Oracle, Versant, and others)
* Levels of abstraction in a DBMS
  + Definitions
    - Data definition language (DDL)
      * Is used to define the external and conceptual schemas
  + Information about the conceptual, external, and physical schemas is stored in the system catalogs
  + Any given database has exactly one conceptual schema and one physical schema but may have several external schemas
  + Conceptual schema
    - Describes the stored data in terms of the data model of the DBMS
    - Describes all relations that are stored in the database
  + Physical schema
    - Specifies additional storage details
    - Summarizes how the relations describe in the conceptual schema are actually stored on secondary storage devices
    - Decisions about the physical schema are based on an understanding of how the data is typically accessed
  + External schema
    - Allow data access to be customized and authorized at the level of individual users or groups of users
    - Each external schema consists of a collection of one or more views and relations from the conceptual schema
      * View is conceptually a relation but the records in a view are not stored in the DBMS
    - Guided by end user requirements
* Data independence
  + Application programs are insulated from changes in the way the data is structured and stored
    - Achieved through use of three levels of data abstraction
    - Relations in the external schema are in principle generated on demand from the relations corresponding to the conceptual schema
    - If the underlying data is reorganized (Conceptual schema is changed (the definition of a view relation can be modified so that the same relation is computed as before
  + Logical data independence
    - Users can be shielded from changes in the logical structure of the data, or changes in the choice of relations to be stored
    - Faculty\_public & Faculty\_private together contain all the information in Faculty
  + Physical data independence
    - Conceptual schema insulates users from changes in physical storage details
      * Hides details such as how the data is actually laid out on disk, the file structure, and the choice of indexes
* Queries in a DBMS
  + Definition
    - Queries
      * Questions involving the data stored in a DBMS
    - Query language
      * DBMS provides a specialized language in which queries can be posted
    - Relational calculus
      * A formal query language based on mathematical logic
      * Queries in this language have an intuitive, precise meaning
    - Relational algebra
      * A formal query language, based on a collection of operators for manipulating relations, which is equivalent in power to the calculus
  + Efficiency of query evaluation is determined to a large extent by how the data is stored physically
  + Indexes can be used to speed up many queries
  + Data manipulation language (DML)
    - DBMS enables users to create, modify, and query data through DML
    - Query language is only one part of the DML, which also provides constructs to insert, delete, and modify data
* Transaction management
  + DBMS must schedule concurrent accesses to data so that each user can safely ignore the fact that others are accessing the data concurrently
  + Locking protocol
    - Definition
      * Locking protocol
        + A set of rules to be followed by each transaction and enforced by the DBMS to ensure that, even though actions of several transactions might be interleaved, the net effect is identical to executing all transactions in some serial order
      * Lock
        + A mechanism used to control access to database objects
    - Two kinds of locks commonly supported by a DBMS
      * Shared locks
        + Can be held by two different transactions at the same time
      * Exclusive lock
        + Ensures that no other transactions hold any lock on object
* System crashes
  + DBMS maintains a log of all writes to the database
    - Write-Ahead log (WAL)
      * Each write action must be recorded in the log before the corresponding changes is reflected in the database
      * DBMS must be able to selectively force a page in memory to disk
      * Ensure that the change made by a successfully completed transaction are not lost due to a system crash
* DBMS support for concurrency control and recovery
  + Every object that is read or written by a transaction is first locked in shared or exclusive mode, respectively
  + For efficient log maintenance, DBMS must be able to selectively force a collection of pages in main memory to disk
  + Periodic checkpointing can reduce the time needed to recover from a crash
* Structure of a DBMS
  + Definitions
    - Query optimizer
      * Uses information about how the data is stored to produce an efficient execution plan for evaluating the query
    - Execution plan
      * Blueprint for evaluating a query, usually represented as a tree of relational operators
    - Relational operators
      * Serve as the building blocks for evaluation queries posed against the data
    - File and access methods
      * Contains code that implements relational operators
      * Supports the concept of a file, which, in a DBMS, is a collection of pages or a collection of records
      * Heap files, or files of unordered pages, as well as indexes are supported
      * Organizes the information within a page
    - Buffer manager
      * Contains code of the files and access methods layer
      * Brings pages in from disk to main memory as needed in response to read requests
    - Disk space manager
      * Deals with management of space on disk, where the data is stored
      * Higher layers allocate, deallocate, read, and write pages through this layer
    - Transaction manager
      * Ensures that transactions request and release locks according to a suitable locking protocol and schedules the execution transactions
    - Lock manager
      * Keeps track of requests for locks and grants locks on database objects when they become available
    - Recovery manager
      * Is responsible for maintaining a log and restoring the system to a consistent state after a crash
  + 
* People who work with database
  + Database implementors build DBMS software
  + End users wish to store and use data in a DBMS
  + Database application programmers develop packages that facilitate data access for end users using the host or data language and software tools that DBMS vendors provide
    - Are usually not computer professionals
  + Database administrator (DBA)
    - Design of the conceptual and physical schema
      * Decide what relations to store and how to store relations
    - Security and authorization
      * Must ensure that unauthorized data access is not permitted
    - Data availability and recovery from failures
      * Must ensure that if the system fails, users can continue to access as much of the uncorrupted data as possible
  + Database Tuning
    - Is responsible for modifying the database to ensure adequate performance as requirements change

Databases design (Chapter 2)

* Entity-Relation data model (ER model)
  + Definition
    - Describe the data involved in a real-world enterprise in terms of objects and their relationships
    - Provides useful concepts that allow us to move from an informal description of what users want from their database to a more detailed, precise description that can be implemented in a DBMS
    - Used in a phase Conceptual database design
  + Steps of database design process
    - Requirements analysis
      * Must find out what the users want from the database
      * Related to ER model
    - Conceptual database design
      * Information gathered in the requirements analysis step is used to develop a high-level description of the data to be stored in the database, along with the constraints known to hold of this data
      * Carried out using ER model
      * Goal is to create a simple description of the data that closely matches how users and developers think of the data
    - Logical Database design
      * Must choose a DBMS to implement database design
      * Convert an ER schema into a relational database schema
      * Related to ER model
    - Schema refinement
      * To analyze the collection of relations in relational database schema to identify potential problems, and to refine it
    - Physical database design
      * Involve building indexes on some tables and clustering some tables
      * Involve a substantial redesign of parts of the database schema obtained from the earlier design steps
    - Application and security design
      * Must identify the entities and processes involved in the application
      * Must describe the role of each entity in every process that is reflected in some application task
      * Must identify the parts of the database that must be accessible and the parts of the database that must not be accessible
* Entities, attributes, and entity sets
  + Definition
    - Entity
      * An object in the real world that is distinguishable from other objects
      * An entity is described using a set of attributes
    - Entity set
      * Collection of similar entities
      * Entity sets need not be disjoint
      * All entities in a given entity set have the same attributes
    - Key
      * A minimal set of attributes whose values uniquely identify an entity in the set
      * There could be more than one candidate key
        + If so, designate one of them as the primary key
* Relationships and relationship set
  + Definition
    - Relationship
      * Association among two or more entities
      * Must be uniquely identified by the participating entities, without reference to the descriptive attributes
    - Relationship set
      * Collection of a set of similar relationships
        + 
      * Entity sets that participate in a relationship set need not be distinct
        + 
    - Descriptive attributes
      * Are used to record information about the relationship, rather than about any one of the participating entities
    - Instance of relationship set
      * A set of relationships
      * 
* Constraints
  + Key constraints
    - Restriction that each department has at most one manager
    - Restriction is indicated in the ER diagram by using an arrow from Departments to Manages
      * 
        + Implies that each Departments entity appears in at most one Manages relationship in any allowable instance of Manages
        + 
  + Participation constraint
    - Restriction that every department is required to have a manager
    - Total participation
      * Entity set Departments in the relationship set Manages
    - Partial participation
      * Participation that is not total
  + Weak entities
    - Entity in which attributes associates with an entity do not include a key
    - Cam be identified uniquely only by considering some of its attributes in conjunction with the primary key of another entity
      * Identifying owner
    - Restrictions
      * Identifying relationship set of the weak entity set
        + The owner entity set and the weak entity set must participate in a one-to-many relationship set
      * Weak entity set must have total participation in the identifying relationship set
    - 
      * Total participation of weak entity is indicated by linking them with a dark line
      * To underscore weak entity and identifying relationship, draw both with dark lines
      * To indicate partial key for weak entity, underline it using a broken line
        + There may be two instances holding same partial key value
* Class hierarchies
  + Class hierarchy can be viewed in one of two ways
    - 
    - Specialization
      * Process of identifying subsets of an entity set that share some distinguishing characteristic
        + The superclass is defined first
        + The subclasses are defined next
        + Subclass-specific attributes and relationship sets are then added
    - Generalization
      * Consists of identifying some common characteristics of a collection of entity sets and creating a new entity set that contains entities possessing these common characteristics
        + Subclasses are defined first
        + Superclass dis defined next
        + Any relationship sets that involve the superclass are then defined
  + Constraints respect to hierarchy
    - Overlap constraints
      * Determine whether two subclasses are allowed to contain the same entity
      * In the absence of statement “Overlaps”, assume by default that entity sets are constrained to have no overlap
    - Covering constraints
      * Determine whether the entities in the subclasses collectively include all entities in the superclass
      * A characteristic property of generalization hierarchies is that every instance of a superclass is an instance of a subclass
      * In the absence of statement “Cover”, assume by default that there is no covering constraint
  + Reasons for identifying subclasses by specialization or generalization
    - Might want to add descriptive attributes that make sense only for the entities in a subclass
    - Might want to identify the set of entities that participate in some relationship
* Aggregation
  + Aggregation allows us to indicate that a relationship set participates in another relationship set
  + Aggregation is used when we need to express a relationship among relationships
  + 
* Conceptual design with ER model
  + Entity versus attribute
    - Use attribute rather than entity
      * Appropriate if we need to record only one address per employee
    - Create an entity called “Addresses” and to record associations between employees and addresses using a relationship
      * Necessary in two situations
        + Have to record more than one address per employee
        + Want to capture the structure of an address in our ER diagram
  + Entity versus Relationship
    - 
      * Natural approach if assume that a manager receives a separate discretionary budget for each department that he or she manages
    - If the discretionary budget (dbudget) is a sum that covers all departments managed by that employee
      * Introduce new entity set called Managers (Which can be placed below Employees in a ISA hierarchy, to show that every manager is also an employee)
      * Attribute ‘since’ and ‘dbudget’ now descrive a manager entity
      * If each manager have a different starting date for each department while every manager has a budget, ‘dbudget’ is an attribute of Managers, but ‘since’ is an attribute of the relationship set between managers and departments
  + Binary relationship vs Ternary relationship
    - Binary relationship
      * A policy cannot be owned jointly by two or more employees
      * Every policy must be owned by some employee
      * 
    - Ternary relationship
      * Dependents is a weak entity set, and each dependent entity is uniquely identified by taking ‘pname’ in conjunction with the ‘policyid’ of a policy entity (which, intuitively, covers the given dependent)
      * 
  + Aggregation vs Ternary Relationship
    - Choice between using aggregation or a ternary relationship is mainly determined by the existence of a relationship that relates a relationship set to an entity set
    - Choice may also be guided by certain integrity constraints that we want to express
    - Use aggregation rather than a ternary relationship set
      * Constraint that each sponsorship of a project by a department is monitored by at most one employee
    - Use ternary relationship
      * A project can be sponsored by any number of departments
      * A department can sponsor one or more projects
      * Each sponsorship is monitored by one or more employees
      * If we don’t need to record the until attribute of Monitors, then use a ternary relationship
* Conceptual design for large enterprises
  + Important aspect of the design process is the methodology used to structure the development of the overall design and ensure that the design takes into account all user requirements and is consistent
  + Usual approach
    - Steps
      * Requirements of various user groups are considered
      * Any conflicting requirements are somehow resolved
      * Single set of global requirements is generated at the end of the requirements analysis phase
    - Generating a single set of global requirements is a difficult task, but it allows the conceptual design phase to proceed with the development of a logical schema that spans all the data and applications throughout the enterprise
  + Alternative approach
    - Develop separate conceptual schemas for different user groups and then integrate these conceptual schemas
    - Steps
      * Must establish correspondences between entities, relationships, and attributes
      * Must resolve numerous kinds of conflicts
    - This task is difficult in its own right
      * In some situations, schema integration cannot be avoided
* Unified modeling language
  + Encompasses a broader spectrum of the software design process than the ER model
    - Business modeling
      * Goal is to describe the business processes involved in the software application being developed
    - System modeling
      * Understanding of business processes is used to identify the requirements for the software application
    - Conceptual database modeling
      * Create the ER design for the database
    - Physical database modeling
      * UML provides pictorial representations for physical database design choices
    - Hardware system modeling
      * UML diagrams can be used to describe the hardware configuration used for the application
  + Diagrams in UML
    - ‘Use case’ diagrams
      * Describe the actions performed by the system in response to user requests, and the people involved in these actions
      * Specify the external functionality that the system is expected to support
    - Activity diagrams
      * Show the flow of actions in a business process
    - Statechart diagram
      * Describe dynamic interactions between system objects
      * Used in business and system modeling
      * Describe how the external functionality is to be implemented, consistent with the business rules and processes of the enterprise
    - Class diagrams
      * Similar to ER diagrams, although they are more general in that they are intended to model application entities and their logical relationships in addition to data entities and their relationships
    - Database diagrams
      * Show how classes are represented in the database and contain additional details about the structure of the database such as integrity constraints and indexes
    - Component diagrams
      * Describe storage aspects of the database as well as interfaces to applications that access the database
    - Deployment diagrams
      * Show the hardware aspects of the system
  + Term relationship is used differently in UML
    - UML’s relationships are binary
    - Relationship sets with key constraints are usually omitted from UML diagrams
    - Relationship is indicated by directly linking the entity sets involved