Database Concepts

presented by:

Tim Haithcoat
University of Missouri Columbia

Introduction

- Very early attempts to build GIS began from scratch, using limited tools like operating systems & compilers
- More recently, GIS have been built around existing database management systems (DBMS)
  - purchase or lease of the DBMS is a major part of the system's software cost
  - the DBMS handles many functions which would otherwise have to be programmed into the GIS
- Any DBMS makes assumptions about the data which it handles
  - to make effective use of a DBMS it is necessary to fit those assumptions
  - certain types of DBMS are more suitable for GIS than others because their assumptions fit spatial data better

Two ways to use DBMS within a GIS:

- Total DBMS solution
  - all data are accessed through the DBMS, so must fit the assumptions imposed by the DBMS designer
- Mixed solution
  - some data (usually attribute tables and relationships) are accessed through the DBMS because they fit the model well
  - some data (usually locational) are accessed directly because they do not fit the DBMS model

GIS as a Database Problem

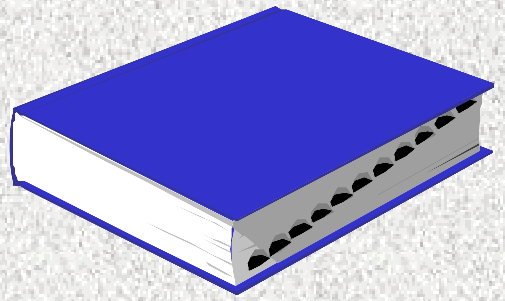
- Some areas of application, notable facilities management:
  - deal with very large volumes of data
  - often have a DBMS solution installed before the GIS is considered
- The GIS adds geographical access to existing methods of search and query
- Such systems require very fast response to a limited number of queries, little analysis
- In these areas it is often said that GIS is a "database problem" rather than an algorithm, analysis, data input or data display problem

Definition

- A database is a collection of non-redundant data which can be shared by different application systems
  - stresses the importance of multiple applications, data sharing
  - the spatial database becomes a common resource for an agency
- Implies separation of physical storage from use of the data by an application program, i.e. program/data independence
  - the user or programmer or application specialist need not know the details of how the data are stored
  - such details are “transparent to the user”

Definition (continued)

- Changes can be made to data without affecting other components of the system, e.g.
  - change format of data items (real to integer, arithmetic operations)
  - change file structure (reorganize data internally or change mode of access)
  - relocate from one device to another, e.g. from optical to magnetic storage, from tape to disk

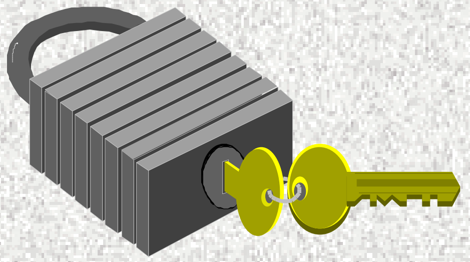


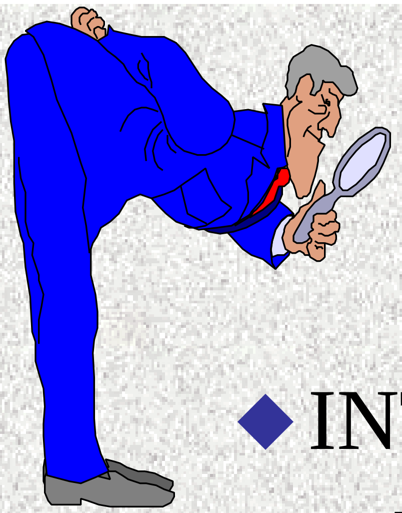
Advantages of a Database Approach

- Reduction in data redundancy
  - shared rather than independent databases
    - reduces problem of inconsistencies in stored information, e.g. different addresses in different departments for the same customer
- Maintenance of data integrity and quality
- Data are self-documented or self-descriptive
  - information on the meaning or interpretation of the data can be stored in the database, e.g. names of items, **metadata**
- Avoidance of inconsistencies
    - data must follow prescribed models, rules, standards

Advantages of a Database Approach (continued)

- Reduced cost of software development
  - many fundamental operations taken care of, however, DBMS software can be expensive to install and maintain
- Security restrictions
  - database includes security tools to control access, particularly for writing



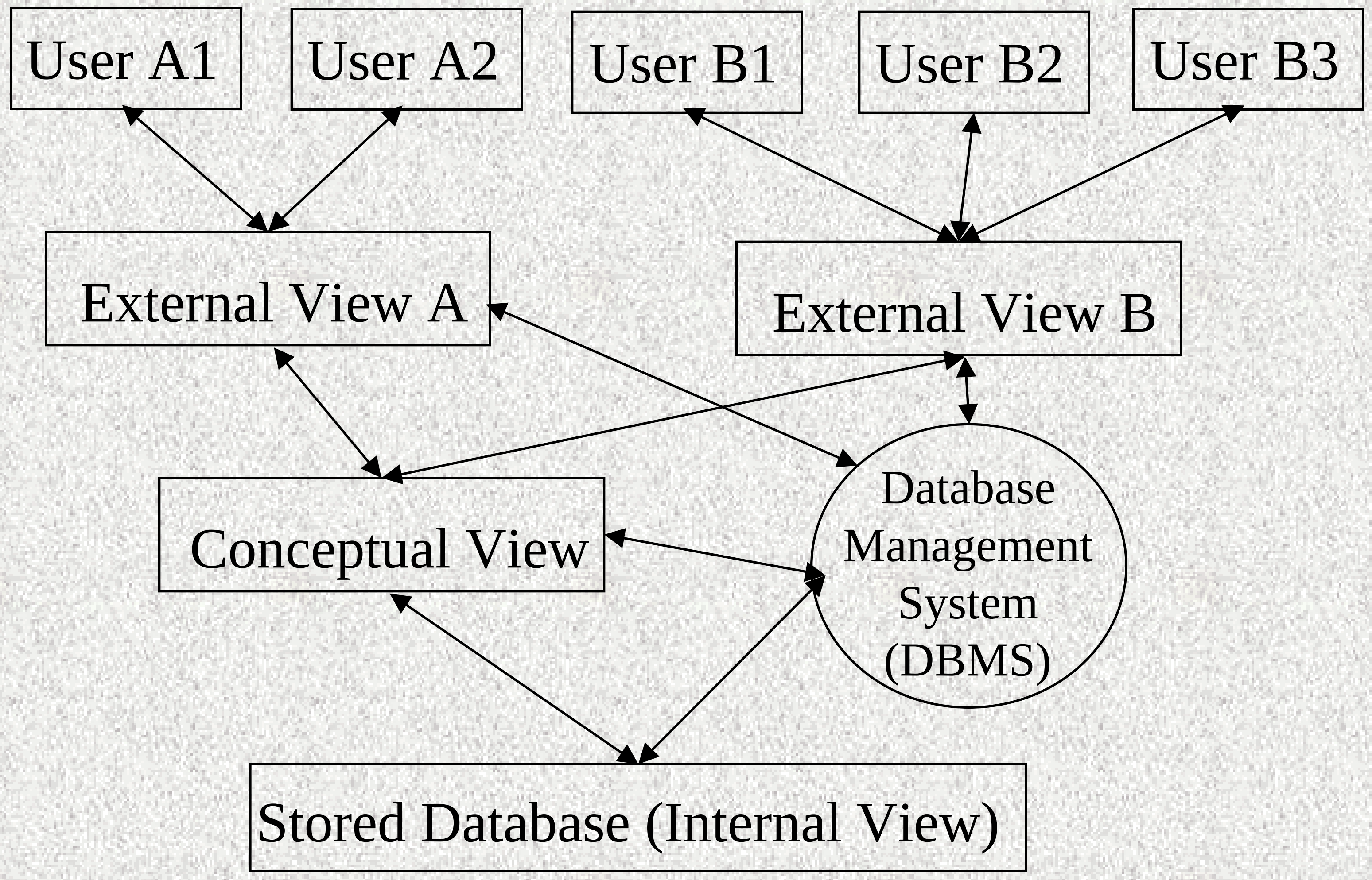


Views of the Database

- INTERNAL VIEW
  - Normally not seen by the user or applications developer
- CONCEPTUAL VIEW
  - Primary means by which the database administrator builds and manages the database
- EXTERNAL VIEW (or Schemas)
  - what the user or programmer sees - can be different to different users and applications

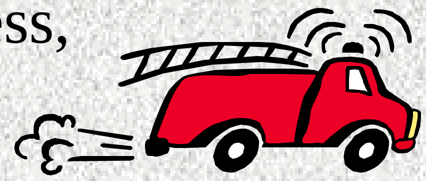
Views of the Database

Adapted from: Date, G.J. 1987. An Introduction to Database Systems, Addison-Wesley. Reading, MA, p. 32



Database Management Systems: *Components*

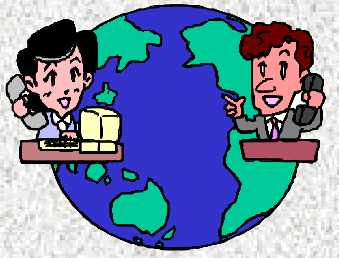
- Data types
    - integer (whole numbers only)
    - real (decimal)
    - character (alphabetic & numeric characters)
    - date
  - more advanced systems may include pictures & images as data types
    - Example: a database of buildings for the fire department which stores a picture as well as address, number of floors, etc.



- Standard Operations
  - Examples: sort, delete, edit, select records

Database Management Systems: *Components* *(Continued)*

- Data definition Language (DDL)
  - The language used to describe the contents of the database
    - Examples: attribute names, data types - “Metadata”
- Data manipulation & Query Language
  - The language used to form commands for input, edit, analysis, output, reformatting, etc.
  - Some degree of standardization has been achieved with SQL (Standard Query Language)



Database Management Systems: *Components* *(Continued)*

- Programming tools
  - Besides commands and queries, the database should be accessible directly from application programs through e.g. subroutine calls
- File Structures
  - The internal structures used to organize the data

Database Management Systems
Types of Database Systems

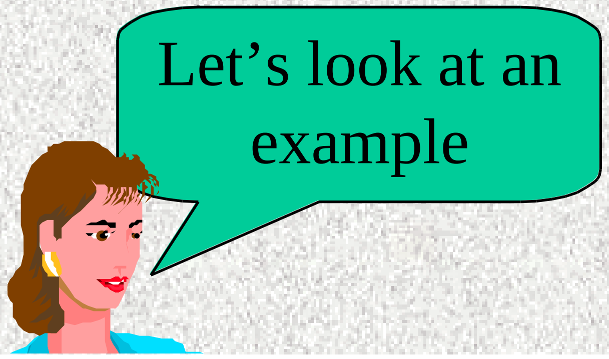
- Several models for databases:
  - Tabular (“flat tire”) - data in single table
  - Hierarchical
  - Network
  - Relational
- The hierarchical, network & relational models all try to deal with the same problem with tabular data:
  - inability to deal with more than one type of object, or with relationships between objects
    - *Example:* database may need to handle information on aircraft, crew, flights, and passengers - four types of records with different attributes, but with relationships between them (“is booked on” between passenger & flight)

Database Management Systems
Types of Database Systems (Continued)

- Database systems originated in the late 1950s and early 1960s largely by research and development of IBM Corporation
- Most developments were responses to needs of business, military, government and educational institutions - complex organizations with complex data and information needs
- Trend through time has been increasing separation between the user and the physical representation of the data - increasing "transparency"

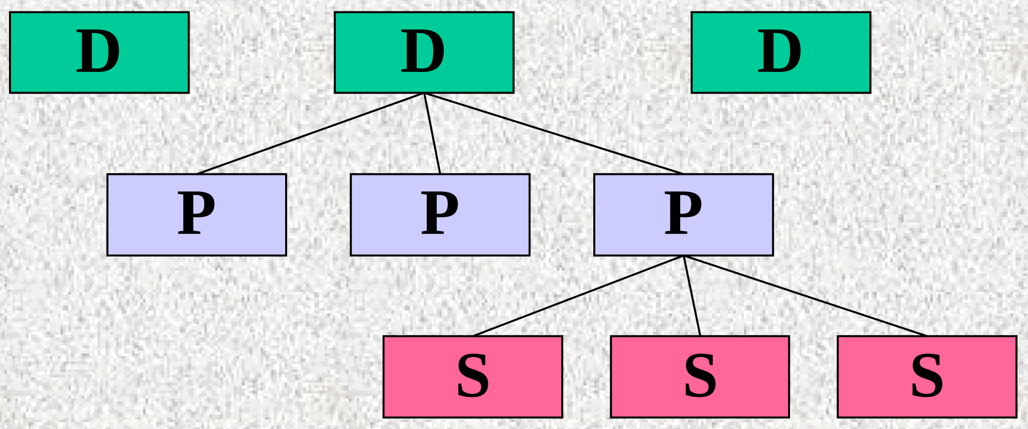
Hierarchical Model

- Early 1960s, IBM saw business world organizing data in the form of a hierarchy
- Rather than one record type (flat file), a business has to deal with several types which are hierarchically related to each other



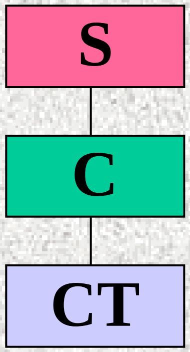
Hierarchical Model

- Example: company has several departments, each with attributes: name of director, number of staff, address
  - Each department requires several parts to make its product, with attributes: part number, number in stock
  - Each part may have several suppliers, with attributes: address, price



Hierarchical Model - Continued

- Certain types of geographic data may fit the hierarchical model well
  - Example: census data organized by state, within state by city, within city by census tract:



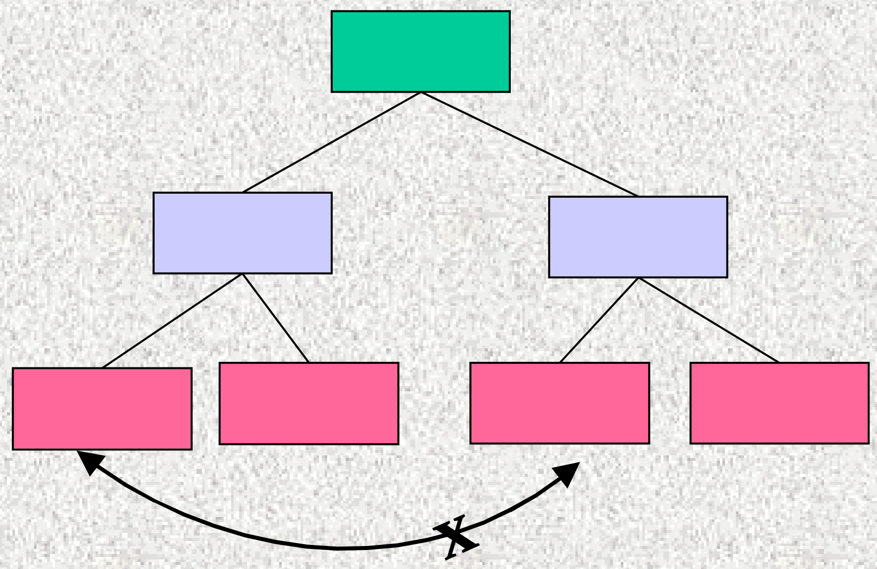
- The database keeps track of different record types, their attributes, and the hierarchical relationships between them
- The attribute which assigns records to levels in the database structure is called the key
  - Example: Is record a department, part or supplier?

Summary of Features

- A set of record "types"
  - Examples: Supplier record type, department record type, part record type
- A set of links connecting all record types in one data structure diagram (tree)
- At most one link between two record types, hence links need not be named
  - For every record, there is only one parent record at the next level up in the tree
    - Example: every county has exactly one state, every part has exactly one department

Summary of Features (continued)

- No connections between occurrences of the same record type
- cannot go between records at the same level unless they share the same parent



Advantages & Disadvantages

- Data must possess a tree structure
  - Tree structure is natural for geographical data
- Data access is easy via the key attribute, but difficult for other attributes
  - In the business case, easy to find record given its type (department, part or supplier)
  - In geographical case, easy to find record given its geographical level (state, county, city, census tract), but difficult to find it given any other attribute
    - Example: find the records with population 5,000 or less

Advantages & Disadvantages (continued)

- Tree structure is inflexible
  - Cannot define new linkages between records once the tree is established
    - Example: in the geographical case, new relationships between objects
  - Cannot define linkages laterally or diagonally in the tree, only vertically
  - The only geographical relationships which can be coded easily are “is contained in” or “belongs to”
- DBMSs based on the hierarchical model (i.e., System 2000) have often been used to store spatial data, but have not been very successful as bases for GIS

Network Model

- Developed in mid 1960s as part of work of CODASYL (Conference on Data Systems Languages) which proposed programming language COBOL (1966) and then network model (1971)
  - Other aspects of database systems also proposed at this time include database administrator, data security, audit trail
- Objective of network model is to separate data structure from physical storage, eliminate unnecessary duplication of data with associated errors & costs

Network Model (continued)

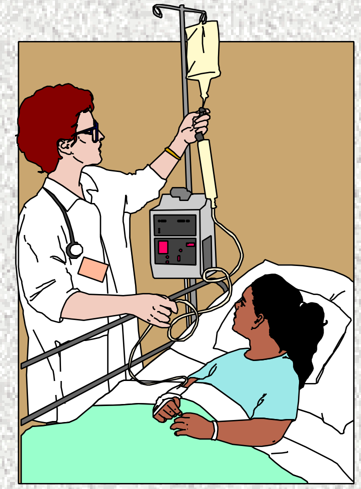
- Uses concept of a data definition language, data manipulation language
- Uses concept of man linkages or relationships
  - An owner record can have many member records
  - A member record can have several owners
    - Hierarchical model allows only 1:n
- Network DBMSs include methods for building and redefining linkages,
  - Example: when patient is assigned to ward

Network Model (continued)

- Example of a network database
  - A hospital database has three record types:
    - Patient: name, date of admission, etc.
    - Doctor: name, etc.
    - Ward: number of beds, name of staff nurse, etc.
  - Need to link patients to doctor, also to ward
  - Doctor record can own many patient records
  - Patient record can be owned by both doctor and ward records

Network Model ~ Restrictions

- Links between record of the same type are not allowed
- While a record can be owned by several records of different types, it cannot be owned by more than one record of the same type
  - Example: patient can have only one doctor, only one ward



Network Model ~ Summary

- The network model has greater flexibility than the hierarchical model for handling complex spatial relationships
- It has not had widespread use as a basis for GIS because of the greater flexibility of the relational model

Relational Model

- The most popular DBMS model for GIS
  - The INFO in ARC/INFO
  - EMPRESS in System/9
  - Several GIS use ORACLE
  - Several PC-based GIS use Dbase III
- Flexible approach to linkages between records comes closes to modeling the complexity of spatial relationships between objects
- Proposed by IBM researcher E.F. Codd (1970)
- More of a concept than a data structure
  - Internal architecture varies substantially from one RDBMS to another

Relational Model ~ Terminology

- Each record has a set of attributes
  - The range of possible values **(domain)** is defined for each attribute
- Records of each type form a table or relation
  - Each row is a record or tuple
  - Each column is an attribute
- Note the potential confusion: a “relation” is a table of records, not a linkage between records

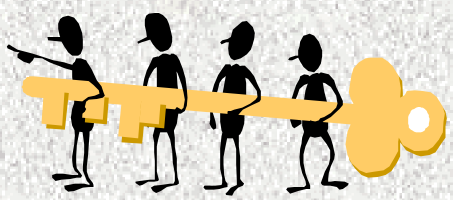
Relational Model ~ Terminology (continued)

- The degree of a relation is the number of attributes in the table
  - 1 attribute is a unary relation
  - 2 attributes is a binary relation
  - N attributes is an n-ary relation
- Examples of relations:
  - Unary: COURSES (Subject)
  - Binary: PERSONS (Name, Address)

    OWNER (Person name, house address)
  - Ternary: HOUSES (address, price, size)

Relational Model ~ Keys

- A key of a relation is a subset of attributes with the following properties:
  - Unique identification
    - the value of the key is unique for each tuple
  - Non-redundancy
    - No attribute in the key can be discarded without destroying the key's uniqueness
  - Example: phone number is a unique key in a phone directory
    - In a normal phone directory the key attributes are last name, first name, street address
    - If street address is dropped from this key, the key is no longer unique (many Smith, John's)



- A prime attribute of a relation is an attribute which participates in at least one key
  - All other attributes are non-prime

Relational Model ~ Normalization

- Concerned with finding the simplest structure for a given set of data
  - Deals with dependence between attributes
  - Avoids loss of general information when records are inserted or deleted
- Consider the first relation (prime attribute underlined):
  - this is not normalized since PRICE is determined by STYLE
  - Problems of insertion and deletion anomalies arise
    - The relationship between ranch & 50,000 is lost when the last of the ranch records is deleted
    - A new relationship (triplex costing 75,000) must be inserted when the first triplex record occurs
- Consider the second relation:
  - Here there are two relations instead of one: One to establish style for each builder, the other price for each style

Relational Model ~ Normalization (continued)

- Several formal types of normalization have been defined - this example illustrates third normal form (3NF), which removes dependence between non-prime attributes
- Although normalization produces a consistent and logical structure, it has a cost in increased storage requirements
  - Some GIS database administrators avoid full normalization for this reason
- A relational join is the reverse of this normalization process, where the two relations HOMES2 and COST are combined to form HOMES1

Advantages and Disadvantages

- The most flexible of the database models
- No obvious match of implementation to model - model is the user's view, not the way the data is organized internally
- Is the basis of an area of formal mathematical theory
- Most RDBMS data manipulation languages require the user to know the contents of relations, but allow access from one relation to another through common attributes

Example

- Given two relations:
  - PROPERTY (ADDRESS, VALUE, COUNTY_ID)
  - COUNTY   (COUNTY_ID, NAME, TAX_RATE)
- To answer the query "what are the taxes on property x" the user would:
  - Retrieve the property record
  - Link the property and county records through the common attribute COUNTY_ID
  - Compute the taxes by multiplying VALUE from the property tuple with TAX_RATE from the linked county tuple

- Setting up and maintaining a spatial database requires careful planning, attention to numerous issues
- Many GIS were developed for a research environment of small databases
  - Many database issues like security not considered important in many early GIS
  - Difficult to grow into an environment of large, production-oriented systems

Databases for Spatial Data

- Many different data types are encountered in geographical data
  - examples: pictures, words, coordinates, complex objects
- Very few database systems have been able to handle textual data
  - Example: descriptions of soils in the legend of a soil map can run to hundreds of words
  - Example: descriptions are as important as numerical data in defining property lines in surveying - “metes and bounds” descriptions

Databases for Spatial Data (continued)

- Variable length records are needed, often not handled well by standard systems
  - Example: number of coordinates in a line can vary
  - This is the primary reason why some GIS designers have chosen not to use standard database solutions for coordinate data, only for attribute tables

Databases for Spatial Data (continued)

- Standard database systems assume the order of records is not meaningful
  - In geographical data the positions of objects establish an implied order which is important in many operations
    - Often need to work with objects that are adjacent in space, thus it helps to have these objects adjacent or close in the database
    - Is a problem with standard database systems since they do not allow linkages between objects in the same record type (class)

Databases for Spatial Data (continued)

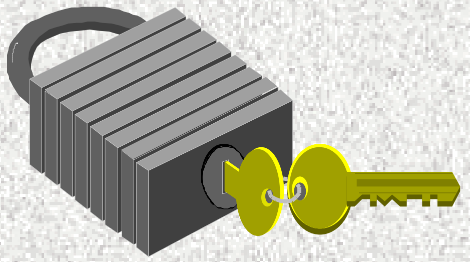
- There are so many possible relationships between spatial objects, that not all can be stored explicitly
  - However, some relationships must be stored explicitly as they cannot be computed from the geometry of the objects
    - Example: existence of grade separation
- The integrity rules of geographical data are too complex
  - Example: the arcs forming a polygon must link into a complete boundary
  - Example: lines cannot cross without forming a node

Databases for Spatial Data (continued)

- Effective use of non-spatial database management solutions requires a high level of knowledge of internal structure on the part of the user
  - Example: user may need to be aware that polygons are composed of arcs, and stored as are records, cannot treat them simply as objects and let the system take care of the internal structure
  - users are required to have too much knowledge of the database model, cannot concentrate on knowledge of the problem
  - Users may have to use complex commands to execute processes which are conceptually simple

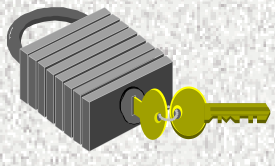
Data Security

- Many systems for small computers, and systems specializing in geometric and geographical data, do not provide functionality necessary to maintain data integrity over long periods of time.



Data Security

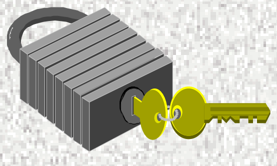
Integrity Constraints



- Integrity constraints: rules which the database must obey in order to be meaningful
  - Attribute values must lie within prescribed domains
  - Relationships between objects must not conflict
    - Example: “Flows into” relationship between river segments must agree with “is fed by” relationship
  - Locational data must not violate rules of planar enforcement, contours must not cross each other, etc.

Data Security

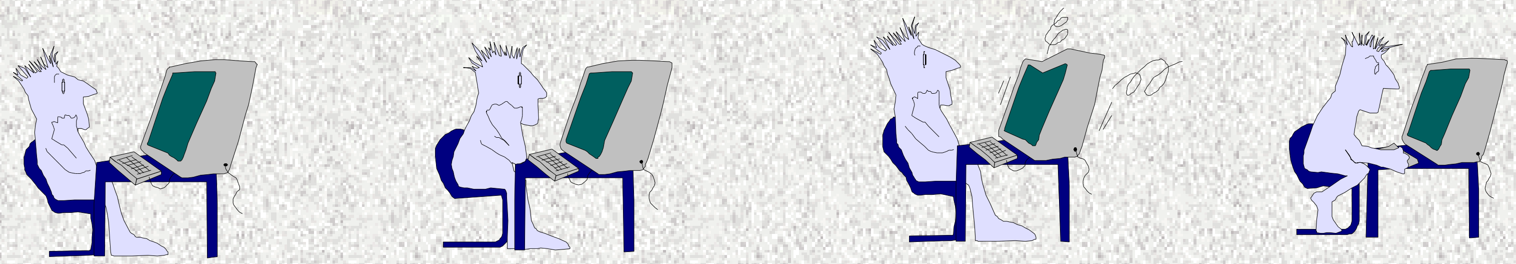
Transactions



- Transactions may include:
  - Modifications to individual data items
  - Addition or deletion of entire records
  - Addition or deletion of attributes
  - Changes in schema (external views of the database)
    - Example: addition of new tables or relations, redefinition of access keys
- All of the updates or modifications made by a user are temporary until confirmed
  - System checks integrity before permanently modifying the database ("posting" the changes to the database)
  - Updates and changes can be abandoned at any time prior to final confirmation

Concurrent Users

- In many cases more than one user will need to access the database at any one time
  - This is a major advantage of multi-user systems & networks
- However, if the database is being modified by several users at once, it is easy for integrity constraints to be violated unless adequate preventative measures exist



Concurrent Users (continued)

- Changes may interfere and produce loss of integrity
  - Example: user B may change an object while user A is processing it
    - The results will not be valid for either the old or the new version of the object
  - Example: a dispatching system
    - Operator A receives a fire call, sends a request to fire station 1 to dispatch a vehicle, waits for fire station to confirm
    - Operator B receives a fire call after A’s call, but before A confirms the dispatch
    - Result: both A & B request a dispatch of the same fire truck
    - Solution: “lock” the first request until confirmed

Concurrent Users (continued)

- Automatic control of concurrent use is based on the transaction concept
  - The database is modified only at the end of a transaction
  - Concurrent users number see the effects of an incomplete transaction
  - Interference between two concurrent users is resolved at the transaction level

3 Types of Concurrent Access

Unprotected:

Applications may retrieve & modify concurrently

In practice, no system allows this, but if one did, system should provide a warning that other users are accessing the data

Protected:

Any application may retrieve data, but only one may modify it

Example: User B should be able to query the status of fire trucks even after user A has placed a "hold" on one

Exclusive:

Only one application may access the data

Check-out/check-in

- In GIS applications, digitizing and updating spatial objects may require intensive work on one part of the database for long periods of time
  - Example: digitizer operator may spend an entire shift working on one may sheet
  - Work will likely be done on a workstation operating independently of the main database
- Because of the length of transactions, a different method of operation is needed

Check-out/Check-in (continued)

- At beginning of shift, operator "checks out" an area from the database
- At end of work, the same area is "checked in", modifying and updating the database
- While an area is checked out, it should be "locked" by the main database
  - This will allow other users to read the data, but not to check it out themselves for modification
  - This resolves problems which might occur

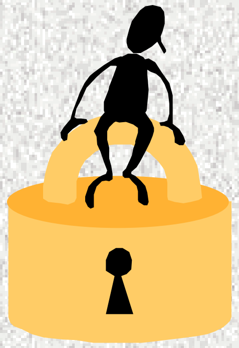
Check-out/Check-in (continued)

- Example:
  - user A checks out a sheet at 8:00a.m. & starts updating
  - User B checks out the same sheet at 9:00 a.m and starts a different set of updates from the same base
  - If both are subsequently allowed to check the sheet back in, then the second check-in may try to modify an object which no longer exists

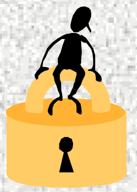
- The area is unlocked when the new version is checked in and modifies the database
- The amount of time required for check-out and check-in must be no more than a small part of a shift

Determining Extent of Data Locking

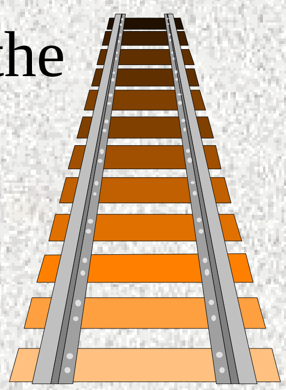
- How much data needs to be locked during a transaction?
  - Changing one item may require other changes as well, (i.e., in indexes)
  - In principle all data which may be affected by a transaction should be locked
  - It may be difficult to determine the extent of possible changes



Determining Extent of Data Locking (continued)



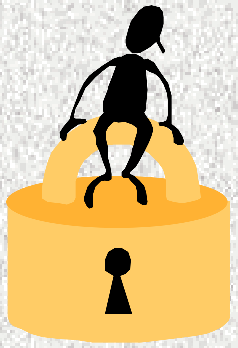
- Example in GIS:
  - User is modifying a map sheet
  - Because objects on the sheet are "edgematched" to objects on adjacent sheets, contents of adjacent sheets may be affected as well
    - Example: if a railroad line which extends to the edge of a map sheet is deleted, should its continuation on the next sheet be affected? If not, the database will no longer be effectively edgematched



  - Should adjacent sheets also be locked during transaction?

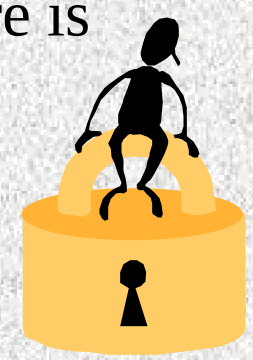
Determining Extent of Data Locking (continued)

- Levels of data locking:
  - Entire database level
  - "view" level
    - Lock only those parts of the database which are relevant to the application's view
  - Record type level
    - Lock an entire relation or attribute table
  - Record occurrence level
    - Lock a single record
  - Data item level
    - Lock only one data item



Determining Extent of Data Locking (continued)

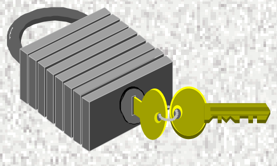
- Deadlock
  - Is when a request cannot continue processing
  - Normally results from incremental acquisition of resources
  - Example: request A gets resource 1, request B gets resource 2
    - Request A now asks for resource 2, B asks for resource 1
    - A and B will wait for each other unless there is intervention



Determining Extent of Data Locking (continued)

- Deadlock (continued)
  - Example:
    - user A checks out an area from a spatial database, thereby locking the contents of the area and related contents
    - User B now attempts to check-out - some of the contents of the requested area have already been locked by A
    - Therefore, the system must unlock all of B’s requests and start again - B will wait until A is finished
    - This allows other users who need the items locked by B to proceed
    - However, this can lead to endless alternating locking attempts by B and another user - the “accordion” effect as they encounter collisions & withdraw
    - It can be very difficult for a DBMS to sense these effects and deal with them

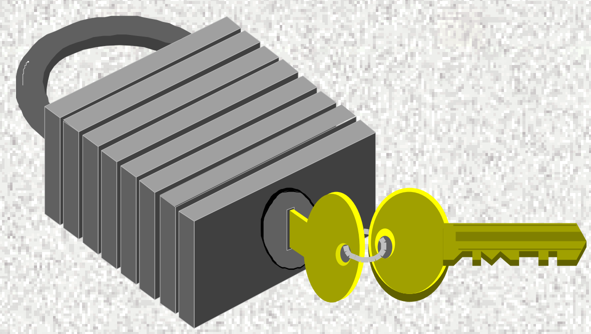
Security Against Data Loss



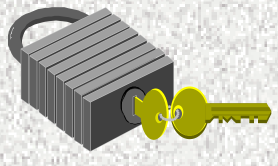
- The cost of creating spatial databases is very high, so the investment must be protected against loss
  - Loss might occur because of hardware or software failure
- Operations to protect against loss may be expensive, but the cost can be balanced against the value of the database
- Because of the consequences of data loss in some areas (air traffic control, bank accounts) very secure systems have been devised

Security Against Data Loss (continued)

- The database must be **backed up** regularly to some permanent storage medium (e.g. tape)
  - All transactions since the last backup must be saved in case the database has to be regenerated
    - Unconfirmed transactions may be lost, but confirmed ones must be saved



Two Types of Failure



- Interruption of the database management system because of operating errors, failure of the operating system or hardware, or power failures
  - These interruptions occur frequently - once a day to once a week
  - Contents of main memory are lost, system must be “rebooted”
  - Contents of database on mass storage device are usually unaffected
- Loss of storage medium, due to operating or hardware defects (“head crashes”), or interruption during transaction processing
  - These occur much less often, slower recovery is acceptable
  - Database is regenerated from most recent backup, plus transaction log if available

Unauthorized Use

- Some GIS data is confidential or secret
  - Examples: tax records, customer lists, retail store performance data
- Contemporary system interconnections make unauthorized access difficult to prevent
  - Example: “virus” infections transmitted through communication networks

Unauthorized Used (continued)

- Different levels of security protection may be appropriate to spatial databases:
  - Keeping unauthorized users from accessing the database - a function of the operating system
  - Limiting access to certain parts of the database
    - Example: census users can access counts based on the census, but not the individual census questionnaires (note: Sweden allows access to individual returns)
  - Restricting users to generalized information only
    - Example: products from some census systems are subjected to random rounding - randomly changing the last digit of all counts to 0 or 5 - to protect confidentiality

Final Thought

Flexibility, complexity of many GIS applications often makes it difficult to provide adequate security