**Title**
Unit 44 - Database Concepts II



**Authors**
Unit 44, CC in GIS
White, Gerald

**Publication Date**
1990

Peer reviewed

# UNIT 44 - DATABASE CONCEPTS II

UNIT 44 - DATABASE CONCEPTS II

Compiled with assistance from Gerald White, California State University, Sacramento



UNIT 44 - DATABASE CONCEPTS II

Compiled with assistance from Gerald White, California State University, Sacramento

## A. INTRODUCTION

- 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, e.g. pictures, words, coordinates, complex objects
- very few database systems have been able to handle textual data
  - e.g. descriptions of soils in the legend of a soil map can run to hundreds of words
  - e.g. descriptions are as important as numerical data in defining property lines in surveying - "metes and bounds" descriptions
- variable length records are needed, often not handled well by standard systems
  - e.g. 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
- 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)
- 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, e.g. existence of grade separation at street crossing
- the integrity rules of geographical data are too complex
  - e.g. the arcs forming a polygon must link into a complete boundary
  - e.g. lines cannot cross without forming a node
- effective use of non-spatial database management solutions requires a high level of knowledge of internal structure on the part of the user
  - e.g. user may need to be aware that polygons are composed of arcs, and stored as arc 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

## The relational model in GIS

- the relational model captures geographical reality through a set of tables (relations) linked by keys (common fields or attributes)
  - each table contains a set of records (tuples)

- tables are normalized to minimize redundancy of information, maximize integrity

- in general, the relational model is a convenient way to represent reality
  - each table corresponds to a set of real-world features with common types of attributes
  - the user needs to know which features are stored in which tables

- however the relational model has certain deficiencies for spatial data
  - many implementations (e.g. ARC/INFO) store only the attribute tables in the relational model, since it is less straightforward to store the geometrical descriptions of objects - such systems have been called "hybrid"
  - most spatial operations are not part of the standard query language of RDBMSs, e.g. find objects within a user-defined polygon, e.g. overlay, e.g. buffer zone generation
  - the relational model does not deal easily and efficiently with the concept of complex objects (objects formed by aggregating simple objects) - this concept is more compatible with the hierarchical data model

## B. 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

### Integrity constraints

- integrity constraints are rules which the database must obey in order to be meaningful
  - attribute values must lie within prescribed domains
  - relationships between objects must not conflict, e.g. "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.

### 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)
    - e.g. 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

## C. 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 and 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

- changes may interfere and produce loss of integrity
  - e.g. 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
  - e.g. 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 may be that both A and B request a dispatch of the same fire truck
    - solution should be to "lock" the first request until confirmed

- automatic control of concurrent use is based on the transaction concept
  - the database is modified only at the end of a transaction
  - concurrent users never see the effects of an incomplete transaction
  - interference between two concurrent users is resolved at the transaction level

Three types of concurrent access

- unprotected - applications may retrieve and 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
  - e.g. 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

Checkout/checkin

- in GIS applications, digitizing and updating spatial objects may require intensive work on one part of the database for long periods of time
  - e.g. digitizer operator may spend an entire shift working on one map 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

- 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
  - e.g. user A checks out a sheet at 8:00 am and starts updating
    - user B checks out the same sheet at 9:00 am 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 checkin 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 checkout and checkin 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, e.g. 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

- e.g. in a 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
    - e.g. if a railroad line which extends to the edge of the mapsheet 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?

- 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

Deadlock

- is when a request cannot continue processing

- normally results from incremental acquisition of resources
- e.g. 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
- e.g. user A checks out an area from a spatial database, thereby locking the contents of the area and related contents
    - user B now attempts a checkout - 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 and withdraw
    - it can be very difficult for a DBMS to sense these effects and deal with them

## D. 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
- 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 the 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

## E. UNAUTHORIZED USE

- some GIS data is confidential or secret, e.g. tax records, customer lists, retail store performance data

- contemporary system interconnections make unauthorized access difficult to prevent
    - e.g. "virus" infections transmitted through communication networks

- 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
        - e.g. 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
        - e.g. 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

Summary

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

REFERENCES

Standard database texts listed under unit 43

Abel, D.J., 1989. "SIRO-DBMS: a database tool-kit for geographical information systems," International Journal of Geographical Information Systems 3:103-116. An extension of the relational model for spatial data.

Frank, A.U., 1984. "Requirements for database systems suitable to manage large spatial databases," Proceedings, International Symposium on Spatial Data Handling, University of Zurich, pp. 38-60.

Nyerges, T.L., 1989. "Schema integration analysis for the development of GIS databases," International Journal of Geographical Information Systems 3:153-184. Looks at formal procedures for comparing and merging spatial database schemas.

EXAM AND DISCUSSION QUESTIONS

1. In what ways are the database issues of GIS different from those of databases generally?

2. What is meant by data integrity in a spatial database? Give examples.

3. Give examples of the ways in which the integrity of a spatial database can degrade without adequate access controls.

4. Examine the database access controls which exist in any GIS to which you have access.

Would they be adequate for a large, production-oriented agency application?

---

*Last Updated: August 30, 1997.*