* Whenever a large amount of data is changed, it shouldn't directly affect the applications that are currently running for the user.
* section 1 discusses the inadequacies of noninferential, formatted data systems
* section 2 discusses how certain relational operations are applied to redundancy and consistency in the user's model
* data independence - independence of applications & terminal activities from growth in data types & changes in data representation
* data inconsistency - troublesome even in nondeductive systems
* the relational view/model of data is viewed to be superior with respect to the graph or network model
  + it describes data w/ its natural structure (doesn't superimpose additional structure for machine representation purposes)
  + provides basis for high level data language
    - yields max independence b/w programs on one hand & machine representation & organization of data on the other
  + forms basis of treating derivability, redundancy, & consistency of relations
  + permits a clearer evaluation of the scope and relative merits of competing representations of data within a single system

SECTION I

* Issue with current data dependencies in present systems
  + how to alter the data representation w/o impairing applications
  + 3 principal kinds of data to be removed
    - ordering dependence
      * data are ordered accordingly
      * con: changing ordering can affect how the application runs
    - indexing dependence
      * improves responses to queries & updates + slowing down responses to insertions & deletions
      * index = redundant component of data representation
      * different approaches to indexing
        + TDMS - unconditional indexing on all attributes
        + IMS - allows user to choose whether to not index or index primary key only
        + IDS - allows file designers to select attributes to be indexed & incorporate indices into file structs. through additional chains
    - access path dependence
      * provide tree structured files or network models of data
        + if there are changes, can impair applications that work with these

solution is that once a user access path is defined, it shouldn't be made obsolete until all applications using that path are obsolete as well (this isn't practical at all b/c the size of the data bank will become too big)

* Relational view of data
  + Ordering of columns matter b/c of possibility of identical headings
    - component - relation depicted
    - ternary relation
      * first 2 domains are called part
      * 3rd domain called quantity
    - component (x, y, z)
      * x is a component or subassembly of part y
      * z units of part x are needed to assemble 1 unit of part y
    - current information systems fail to provide data representations for relations that have 2 or more identical domains (IMS/360)
  + Users shouldn't have to deal with remembering the domain ordering of any relation
    - should instead deal with relationships
    - w/ 2 or more identical domains, must be identified by a distinctive role name
    - in conclusion, should interact w/ relational model of data consisting of a collection of time varying relationships & only need to know the names
  + relations on simple domains
    - 1 domain or combo of domains of a given relation
      * primary key
        + redundant if either single domain or combo where no participating single domains is superfluous in uniquely identifying each element
* normal form
  + normalization (graph)
* linguistic aspects