CSCI 403: Databases
1 - Introductory Material

Definitions

What is data/what are properties of data?
Your answers in class:

- information
- meaningful
- organized
- numbers, strings, etc.

What is a database/what are some defining characteristics of a DBMS?
Your answers in class:

- organized data
- easy retrieval
- modifiable
- relate data to other data
- extract meaning from data

Properties of the Modern Database

Self-describing

A DBMS allows us to define a database structure and store this definition together with the data. The data description is often called "meta-data". In DBMS terminology, the meta-data is stored in the database catalog and records details of all object types in the database (using object here loosely, not in the OOP sense). Key advantage here is that the database software is general-purpose and will work equally well for one application as for another.

Consider file-based system instead. Pre-xml/JSON, or even pre .ini file, no standard way to store data specific to an application; would often store data using the byte layout of the structs in the programming language. Each program would process its files and no others.

Program-data separation

Another benefit of self-description is that programs and the structure of data can be changed independently. In the file example, a change to the structure immediately requires both a change to program code and a rewrite of all data files from the old format to the new. With a DBMS, you can typically modify the database structure (e.g., by adding a field or a relationship) without recoding existing applications (obviously some changes will break software - deleting record types/relationships, some modifications). Beyond that, software can be written to recognize and work with modified structures by reading the catalog (meta-data), so for instance, adding a field can actually enhance result in a general-purpose query program.

Data abstraction

Structure of data storage on storage medium completely abstracted away - application programmer does not need to know!!!

Network multiuser access

A key feature from early days of DBMS is the ability to access database via multiple programs (or by multiple users) at the "same" time. This requires transaction control to ensure that updates from one user are not inconsistent with updates from another user (e.g., when one user changes data, the modification is to a current view of the data rather than an out-of-date view). This tends to go together with network access (since without it, you don't tend to have multiple users).

Client-server architecture

Client-server architecture arises organically from all of the above concerns. Since the database is self-describing and independent of application programs, it makes sense to build software devoted solely to providing database services. Add in network multi-user access, it is natural for applications to exist as separate client software, making a limited set of calls to execute DB functions on the server.

Early Models

Pre-1960, data was stored on sequential access media (tape, punch cards, punched tape). Data storage was typically application specific. No standard, general purpose system for storing, indexing, retreiving data. First general purpose DBMSes date after introduction of direct access (e.g, disk) storage. Term "data-base" dates to 1962 (according to OED via Wikipedia).

Charles Bachman, GE 1964. IDS (Integrated Data Store) Bachman (1924 - 2017) is the 1973 Turing Award winner. IDS is the original network model database system, which was later standardized by CODASYL.

CODASYL stands for "Conference/Committee on Data Systems Languages", a consortium founded in 1959 to develop a standard programming language. Two notable achievements: COBOL standard, and network (aka CODASYL) database standard. In 1965 formed List Processing Task Force to develop extensions to COBOL for managing collections of records (with specific reference to IDS network model). In 1967 was renamed to "Data Base Task Group" (DTBG). 1969 - first network model standard. Defines DDL and DML (extensions to COBOL). Movement towards language independence in 1971. A number of vendors created DBMSes following the CODASYL standard, at least one of which is still sold today.

Bachman also developed (1965) an early transaction control system (allowing for multiuser network access).

Network model: flexible & powerful graph-based storage of records. Essentially composed of "records" and "sets". Records organized into named types defining the attributes for a particular record name. Set types define relationships between records. Specifically, a set type defines an "owner-member" relationship between two record types. An owner record is associated by a set with 0 or more member records, which are all of the same type. All records assigned a unique key value (tied to storage in system, allows for essentially direct access to record). Navigation between owner and members was via record id linkage (circular list anchored by owner). Power and flexibility in the system due to fact that records could participate in multiple sets (of multiple types), allowing for complex graph structures.

IBM 1966-68 IMS (Information Management System), for the Apollo space program (bill of materials tracking for Saturn V rocket). The first hierarchical model database system. IMS still sold today. In hierarchical model, records form a tree structure; each record can have at most one parent, but multiple child records. (Think of file systems.)

Both IDS and IMS were/are examples of navigational database systems. Access to a record is primarily predicated on knowing the record's unique key, and data retrieval follows linkages (like pointers) from parent to child (or to sibling in the case of network model).

The Relational Model

E. F. Codd (1923 - 2003). Employed by IBM, in 1970 publishes "A Relational Model of Data for Large Shared Data Banks". Internal pressure at IBM was to preserve revenue stream from IMS. IBM eventually developed a product based on Codd's ideas, called System R. System R project developed SQL (originally as SEQUEL). System R first sold in 1977. Codd wins 1981 Turing award for his work.

Oracle (initially developed 1977-79) is released by Larry Ellison's Relational Software in 1979. World domination soon follows.

INGRES - U. Berkeley project begun in 1973 by Michael Stonebraker & Eugene Wong after reading technical papers from the System R project.

Source code available early on (first open-source DB?) Ingres initially competed with Oracle, but lost partly due to having own query language (QUEL). Project ended in 1985. Some of Ingres developers went on to develop commercial DBs, notably Sybase (which is also progenitor of MS SQL Server).

Postgres - Michael Stonebraker starts project at Berkeley in 1985 to address shortcomings of relational DBMSes of time, particularly inability to define new data types. (Still didn't support SQL until 1994.) Goes fully open source in 1994.

Michael Stonebraker is 2014 Turing Award winner.

Features of Relational Model

We will spend a lot of time on this in future lectures, but basically relational databases move away from pointer-based navigational approach to a more flexible approach based on set theory. Different views of the data can be dynamically created based on relational joins - these do not have to be designed into the database structure from the start. Initial versions were slow compared to navigational DBMSes, but rapidly improved with better indexing schemes, etc., to become dominant. The relational DBMSes dramatically improved data abstraction and program-data separation.

New (Old) Ideas

Flirtation in 1990s with OODBMSes (with rise of OOP). Note that OODB reverts to navigational concepts! OODB concepts were quickly subsumed into mainstream relational products such as Oracle to make the ORDBMS. Relational DBMSes have continued to expand into new data types (e.g., XML, BLOB, GIS) and gain new capabilities. However, the rise of the internet has led to a perceived need for new paradigms.

NoSQL (Not only SQL) databases are not well-defined, but exist in part due to the very large, very rapid, quickly changing data streams created by Internet usage and commerce. Some characteristics:

- Scalability - distributed over many nodes
- Flexible schema - e.g., document databases using JSON - good match to agile development processes - not tied to a fixed structure
- Fault tolerance - survival of single node downtimes

Some of the "new" databases are navigational in nature - graph databases share some similarities with network databases, for instance.

Note that all of these features are also being addressed by Oracle, Postgres in various ways, so whether NoSQL will remain a separate movement is yet to be determined.

Future - Rise of "New-SQL" databases?