The Past, Present, and Future of Computer Database Technology

Todd J.

CS-407 – Database Administration & Programming

Professor Gholam Omidipour

September 11th, 2010

Table of Contents:

Introduction:

The need for proper storage of information has become a paramount endeavor since the earliest period in mankind's history. From the very first cave drawings, to currently breaking news, we have been driven by the desire to record and catalog almost every piece of information. It is with that mindset that imposes on us the fundamental societal goal of consistently improving and enhancing our database technologies.

The Database Is Born:

Prior to 1960, there existed very little in the way of data retrieval or storage similar to the database technology that we use today. It was not until 1961 when a man by the name of Charles Bachman began work on the initial ideology of modern database structures. Born in 1924, his first overall exposure to computers was in 1944 while serving a tour in the Pacific theater. His use of armament computer technology in the military intrigued his curiosity early on. This began a life-long interest in technology that stayed with him throughout his career. After graduating from the University of Pennsylvania with a Masters degree in mechanical engineering, he went to work for Dow Chemicals. It was not until he joined General Electric in 1960 however, that he first created the Integrated Data Store. The IDS, as it was called, became one of the industry's first database management systems. Among some of his other achievements at General Electric was a special version of his database application called *dataBasic*. DataBasic provided database functionality while utilizing the Basic programming language (DuCharme, 2005).

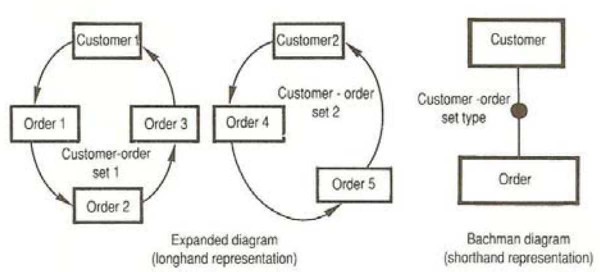

Fig. 1 – The Bachman Diagram
(Wikimedia Commons, http://en.wikipedia.org/wiki/File:Bachman_diagram.jpg)

Bachman went on to work for several other prominent organizations before creating his own corporation. Through his prestigious career, he laid the foundational ground-work for modern database development and that of the network database diagram.

Early Database Usage and Data Models:

Computer usage was fairly uncommon in the private sector prior to the early 1960s. Databases did not exist in any real capacity due to digital storage being so limited and costly. Overwhelmingly, the type of data that *was* stored was often quite basic. This was typically nothing more than data mappings used to aid in various calculations and functions. The government used these data mappings for storage of coordinates and numbers to help with the military and space programs. Prior to the invention of magnetic media, data had to be cataloged through the use of punch cards. As magnetic media and hard disk technology improved and became more common in the early 1960s, so did database usage in big industry.

Fig. 2 – Honeywell 400 – Early Database Storage & Processing
(Creative Pro – Getting Things Done in 1961 http://www.creativepro.com/)

Through the early 1960s, there were quite a number of database designs. Many of these designs had similar topologies however. Despite the multitude of early database technology, there were primarily only two common structures by which most were based. These two types of databases were known as the hierarchical and the network model. Database structures are referred to as models due to the layout of the data diagram schematics of which they are made. The first of these database designs to gain significant traction in the industry was the hierarchal model (DuCharme, 2005).

<u>The Hierarchical Database Model:</u>

The hierarchical data model is a unique data type structure which identifies its data through a side-ways tree-like structure. The pure benefit of the hierarchical data-structure when compared with other database models is in its transactional speed. Most hierarchical database systems are purpose-built databases running on purpose-built environments. This enables transactional data to process in typically half the amount of time necessary than any other database model on an otherwise comparably equipped data-system and hardware. Hierarchical databases make this possible because of the manner in which the data is stored within the system. While data is stored in a tree pattern, each successive child node carries with it the entire redundant data segment through to the root parent node. While exceedingly fast, the trade-off for this speed is the mandatory increased storage requirements. The excess data redundancy requires considerably more storage capacity than other database models, which increases hardware costs (Walters, 1997).

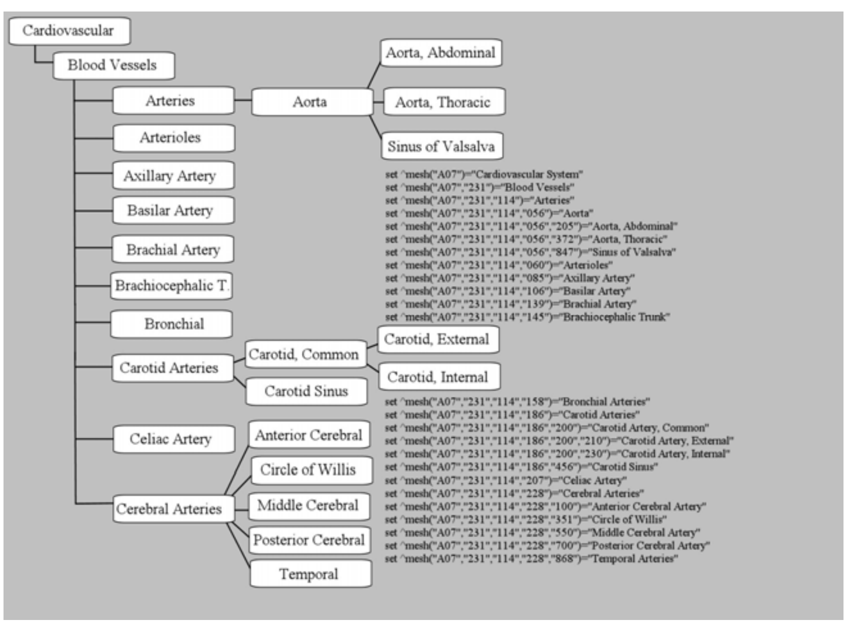

Fig. 3 – Mumps Node Structure
(Mumps MultiDimensional Database, http://www.cs.uni.edu/~okane/mumps.html/)

One relatively unknown but widely used example of this database model is a variation known as MUMPS. The MUMPS acronym stands for the Massachusetts Utility for Multi-Programming Systems. Having first entered production in 1968, it is now considered antiquated by most standards. However, the MUMPS environment and data structure is still commonly used today in a number of corporations and industries. The most common of these uses exist in the medical and banking industry due to the fast transactional times afforded with patient and customer records. The system exists within its own environment and therefore utilizes a specialized mechanism for data retrieval. The language utilized within this environment is also known as Mumps, but is more commonly referred to as the “M” language. In what is known as an interpreted environment, the development and query language are one in the same. This means that all software developed for the environment, database, and terminals exist within the same allocated space as the residing data. A few examples of highly popular MUMPS software include the Antrim System which was designed for patient pathology, and IDX which was designed for medical patient billing (Walters, 1997).

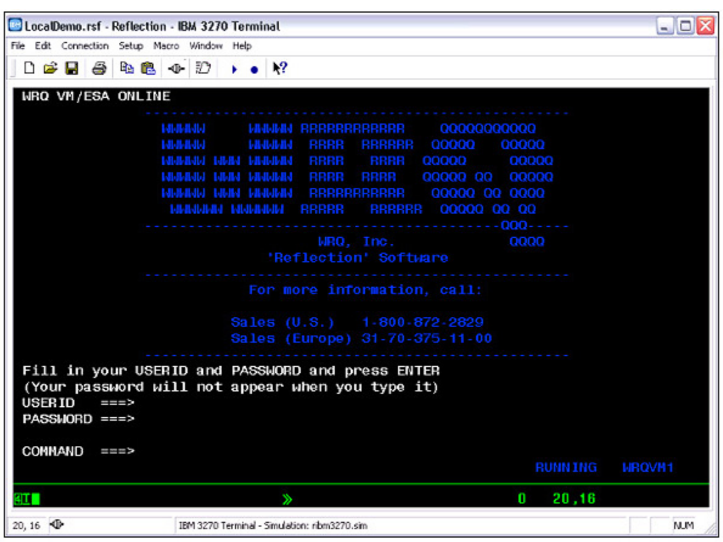

Fig. 4 – IBM Reflections Terminal Services
(IBM Corporation, http://www.ibm.com/)

Because these systems were originally designed for outdated mainframes, they require the use of emulation software to operate on modern servers. Likewise, the end-user requires terminal software such as IBM Reflections to be installed before the environment is accessible. Newer instantiations of this system, such as a product known as Cache, have been designed to integrate with web technology. In addition, some of the newer features available with this product provide the ability for relational mapping and retrieval of hierarchical data with ANSI derived SQL. Cache by InterSystems Corporation even provides the ability to integrate the M language and its database on multiple platforms through the use of M scripting language; however, this database structure is for the most part, still obsolete.

Fig. 5 – InterSystems Cache' Logo
(InterSystems Corporation, http://www.intersystems.com/)

Although the environment and products supported by a hierarchal database continue to remain viable even with today's technology, few if any new corporations are making an investment in this direction. The only corporations that still utilize this technology are existing legacy customers. For most of these companies, they are seeking to upgrade or shift to newer technologies on more common platforms due to the excessive cost associated with legacy technology and the Cache fee structure.

The Network Database Model:

A slight improvement over the hierarchal model, the network model made significant advancements in data structuring. While similar in many respects to the tree-like structure of a hierarchal database, the network model provides for the ability of basic table referencing with what it calls table "arcs." What this essentially means is that, unlike a hierarchal database, the network model allows each data node to have multiple child and parent relationships. A set of data nodes existing under different parent nodes can still relationally be connected to each-other. The hierarchal database by comparison requires each node to have only a single delineation of parent to child drill-down in its structure. This is said to provide "a more natural modeling of relationships between entities" (as cited in Network Model, 2010).

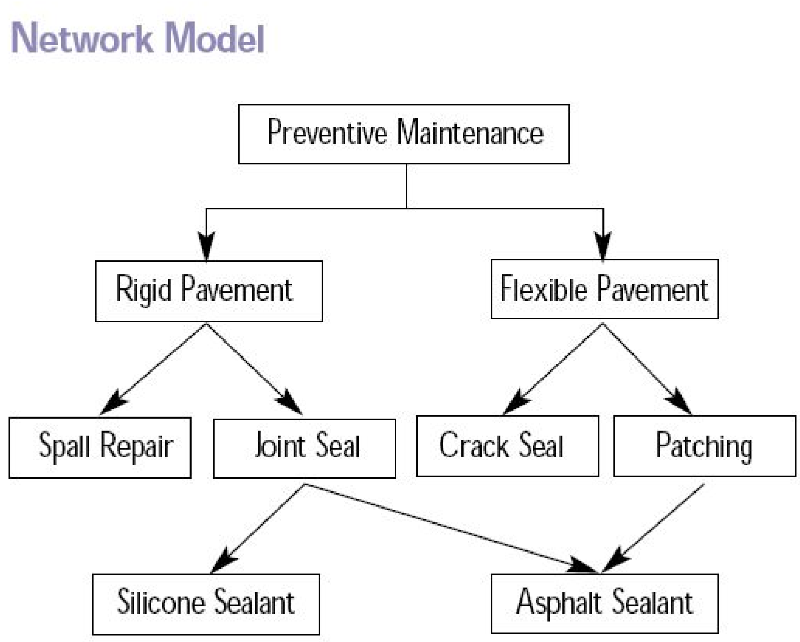

Fig. 6 – The Network Database Model
(WikiMedia, http://en.wikipedia.org/wiki/File:Network_Model.jpg)

Throughout the late 1960s’ and well into the mid 1970s’, the network model became popular primarily for the leading-edge segment of business technology development. The network database model found use and support from a variety of companies and systems, some of which included designs by Univac and DEC. One of the foremost developments utilizing this design was in a system by Honeywell Corporation. This system utilized the then ground-breaking design of the Integrated Data Store DBMS by Charles Bachman. The IDS, as it was known for short, was popular through the mid to late 1960s, particularly with the military and aerospace industry (DuCharme, (2005).

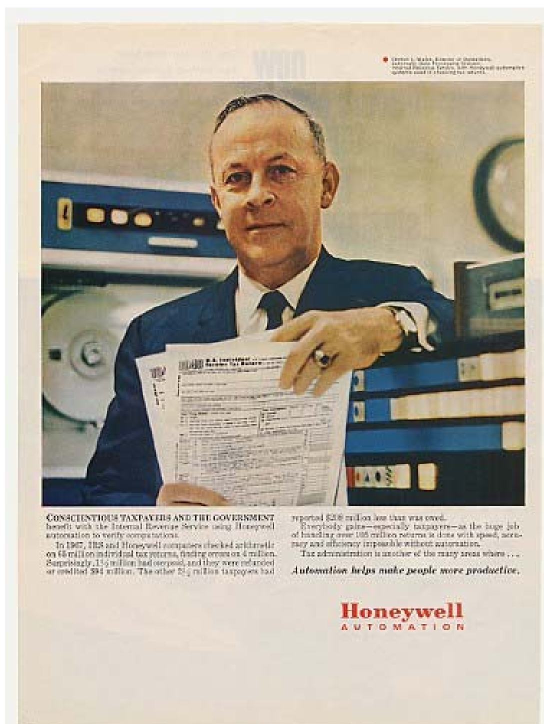

Fig. 7 – A 1967 Honeywell IDS Advertisement
(Honeywell Corporation, http://www.honeywel.com)

Like the hierarchal model however, the network structure is of an older technological design and is no longer being used or implemented in the industry today.

The Relational Database Model:

The relational model is the most popular of the three common database structures, and is also the system most recently developed. The relational model incorporates much of what made the network model popular while necessitating less physical storage requirements. The relational model is substantially more advanced than any other database model in production today (Relational, 2009).

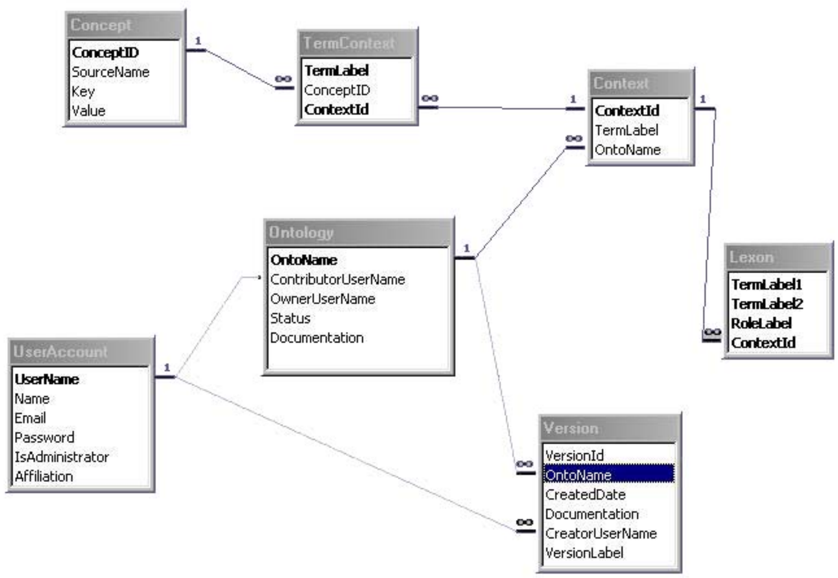

Fig. 8 – Example of a relational table diagram
(Microsoft Access, http://www.Microsoft.com)

What makes the relational model so advanced is in its ability of table referencing. The entire structure of the data is stored differently than in other database model. The data is instead stored directly within tables, with static fields that do not change. With a set of tables in a relational database, each table is likely to contain a series of keys. These keys, either a primary or foreign key, represent the primary fields within the table. The table that holds the highest level of data would then likely exist with a primary key. This primary key could then be used to reference another table with an identically named field.

A foreign key, by comparison, is a secondary field within a more primary table that is used to reference another table where that same field is used as a primary key. In the example of an item inventory table, the *item_id* would be represented as the primary key. The *company_id* within that same table would be defined as the foreign key. This foreign key could likely then refer to a “Company Table” where-by the *company_id* in that table would be the primary key. Utilizing a specialized query language known as SQL, this data can then be manipulated and retrieved in a variety of efficient ways (Loney, 1997). For example, if a database user wanted to see the entire inventory table complete with the company’s information, this would necessitate what is called a “table join.” In this case, the query might look something like this:

```
Select * From tblItems, tblCompany Where tblItems.CompanyID =
tblCompany.CompanyID
```

This query would return all records from the items table complete with the corresponding company information from the company table (Rob, Coronel, 2009).

Relational databases continue to remain the dominant force in today’s industry. Additionally, modern software technology products rely heavily on the ability to manipulate data using this highly versatile query and scripting language. There are dozens of major corporations capitalizing on this market and providing their own database products utilizing this database model and schema. Some of the most well known products include SQL Server by Microsoft, Oracle Server by Oracle Corporation, Interbase by Borland Corporation, and the open-source MySQL. There are even some more compact desktop level databases such as Access by Microsoft, FileMaker Pro by FileMaker Inc., and Paradox by Corel (Chapple, 2010).

The Future of Database Technology:

Unlike the network and hierarchical database models, the design which became the most popular in business was the relational model. This type of database structure is not only one of the most efficient, but has created an entire market full of products which have mushroomed with features in recent years.

Database systems have always provided the ability to store and retrieve data, that was their primary purpose and they did it exceedingly well. However, what more companies consistently seek is the ability to obtain ready-made *information* directly from the database. What used to take multiple developers and entire marketing analyst teams weeks, can now be accomplished in minutes with many of the new features in these modern database products. This technology, known generically as "business intelligence" provides the customer with the ability to compile everything from trend analysis to statistical reporting, all through a sophisticated database interface.

Fig. 9 – Microsoft SQL Server Products
(Microsoft Corporation, http://www.Microsoft.com)

As once explained by Hans Luhn of IBM, business intelligence is "the ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal (Luhn, 1958)."

Although originally coined in 1958, business intelligence or BI as abbreviated, was not as significant or as conceptually advanced before major modern advances in database technology. The biggest hurdle with functional analysis software was typically acquiring and achieving the massive processing resources that were required to run it. With major improvements in hardware cost and performance, many organizations are now seeking BI products as a solid business solution. Because of the growing demand and recent dynamic success with this technology, several companies are beginning to offer BI products either as an enterprise add-on to their own database system, or as third party vendors to existing popular database solutions. The more common of these products are SQL Server Analysis Services and Oracle's Business Intelligence Suite (formerly known as Brio). These products, when integrated into the company's database management system, provide the consumer with the ability to retrieve data in a process known as data mining. By manipulating specific data components known as database cubes, the end-user can produce data results through a variety of different ways (Business Intelligence, 2010). The export data can be displayed through numerous forms of outputs, depending upon the consumer's requirements.

The depth with which a business intelligence solution can be implemented depends upon the skill and resources of the user. More advanced users can create ready-made BI reports on the fly for the end-user. Known as "on-demand" reports, these are typically available through a type of internal enterprise web-portal. Dedicated report developers and programmers can publish these reports on the web portal, making them accessible either to specific individuals or to entire groups. The reports are tied to the data objects, so the information reflected on them are consistent with what exists in the database at the time of processing (Course 2780B, 2005).

More advanced end-users can produce their own ad-hoc reports through Microsoft Excel data integration techniques. These techniques allow the manipulation of data in the form of a spreadsheet simply by adding or subtracting data combinations or results. The data can be displayed as columnar, or through multiple object modules on the report. The modules might be represented on the spreadsheet as that of raw derived data, or in more graphical forms such as a chart or graph.

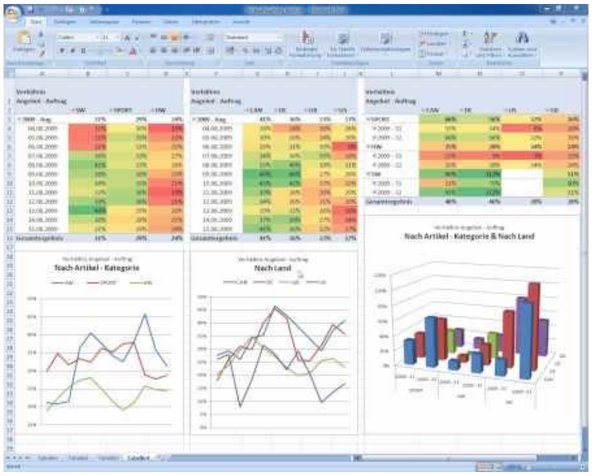

Fig. 10 – Microsoft SQL Server Products
(Microsoft Corporation, http://www.Microsoft.com)

Although business intelligence has existed in theory for many years, the latest technology brings it to the forefront of the business model. Nearly every industry, company, and department stands to gain significant insight into not only the current health and statistics of the organization, but the future of it as well.

Conclusion:

At one point in time, the computer database served no other purpose than as a simple means for storing and retrieving basic data. From an elaborate dedicated mainframe system with multiple terminals, to an advanced but simple desktop suite, the database has become a staple in American and world-wide industry. What was once a relatively simple system, the database has become a highly sophisticated intelligent resource for data input and information retrieval. Database technology continues to grow in leaps and bounds and will remain a dominant necessity in business for decades to come.

References:

DuCharme, Bob. (2005, December 6) *25 years of database history (starting in 1955)* DuCharme's Blog Retrieved September 7, 2010, from http://www.snee.com/bobdc.blog/2005/12/25-years-of-database-history-s-1.html

Wikipedia *Hierarchical model (database)*. Retrieved September 2, 2010, from http://wikipedia.org/wiki/Hierarchical_database_model

Walters, R. (1997) *M programming: a comprehensive guide*. Boston, MA: Digital Press.

Wikipedia *Network model (database)*. Retrieved September 8, 2010, from http://wikipedia.org/wiki/Network_database_model

Loney, Kevin (1997) *Oracle dba handbook 7.3 edition*. Berkeley, CA: Osborne McGraw-Hill.

Rob, P, & Coronel, C. (2009). *Database systems: design, implementation and management*. Boston, MA: Course Technology.

Chapple, Mike. (2010) *Guide picks - top 5 desktop databases* About.COM : Databases Retrieved Sept. 8, 2010, from http://databases.about.com/library/aatp101401a.htm

Wikipedia *Relational model (database)*. Retrieved September 11, 2010, from http://wikipedia.org/wiki/Relational_model

Luhn, Hans. (1958) A Business Intelligence System. *IBM Journal*, Retrieved from http://www.research.ibm.com/journal/rd/024/ibmrd024h.pdf

Wikipedia *Business Intelligence* Retrieved October 2, 2010, from http://en.wikipedia.org/wiki/Business_intelligence

(2005) Course 2780B maintaining a microsoft sql server 2005 database. Redmond, WA: Microsoft Learning.