1987

A relational database in support of a CIM system /

Ronald Habakus
Lehigh University

A RELATIONAL DATABASE IN SUPPORT OF A CIM SYSTEM

by

Ronald Habakus

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Manufacturing Systems Engineering

Lehigh University

1987

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science

May 18, 1987
(date)

Roger N. Nagel
Professor in Charge

Roger N. Nagel
MSE Program Director

S. E. Kane
Chairman of Department

ACKNOWLEDGMENTS

There are several people who deserve recognition for the work performed here. First, my thesis adviser, Dr. Roger N. Nagel, is to be thanked for his guidance in the directions the thesis took. Second, a word of thanks for Mrs. Susan Swartz for her incites and kind editing support. Third, to fellow MSE student Michael Lawrence Connolly, who's unabashed approach to life provided me with a unique sense of inspiration, may happiness always continue to flow your way and your sense of adventure never die. Fourth, and most importantly, I want to thank my parents Louise M. and William S. Habakus for their never ending patience and support. Their multitudes of help have always provided me with the opportunities to choose the paths that I desired.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

As manufacturing markets become increasingly competitive, the application of computer technology on the shop floor becomes more and more of a strategic issue. Computer Integrated Manufacturing (CIM) is a philosophy advocating the use of computers to integrate the functions involved with manufacturing. The most important aspect for successfully integrating these functions is to establish effective intercommunication that provides the necessary data when it is needed.

Database management systems (DBMS's) are data processing tools that furnish a control over data which can be useful in CIM systems. There are many types of DBMS's commercially available, but the systems based on the relational database model are the most advanced and user friendly.

This study examines the implementation of a relational database system used for labor, material, and job tracking within a small CIM system on a shop floor machining cell. Database technology is explained, highlighting its importance with specific focus on relational database technology. The ORACLE relational DBMS used for the

application is reviewed, and the development of the database is outlined. Lastly, the system is critiqued, discussing its accomplishments, weaknesses and limitations, and areas for future study are suggested.

The database system drastically reduced the number of manual data flows on the shop floor, and provided productivity improvements in the areas of employee time tracking, job time reporting, material tracking, quality assurance, and checking job status'. Weaknesses found were cost, complexity, and system vulnerability to failure. The application's limitations included a lack of sufficient hardware on the shop floor, the possibility for entering erroneous data, restricted capabilities for job status checking, and a lack of integration with other systems.

Relational databases were found to be extremely powerful tools for data control, and their use in CIM was projected to rapidly increase.

I INTRODUCTION / BACKGROUND

I.1 - Introduction

Computer based information technology is being researched and developed today at a rate faster than ever before. Along with these new advances in technology come new opportunities to be realized in all aspects of business. Manufacturing too is receiving higher levels of attention than it has in the recent past. The marriage of information technology to the field of manufacturing shows great promise for reaping benefits.

Databases represent a major development in data processing due to the increase in data handling efficiency and flexibility they provide. While databases are currently used in manufacturing, their application is not truly extensive. Of the three main types of databases, hierarchical, network and relational, the relational database model is the most recent and consequently the least used in manufacturing. As relational databases become better known, this is expected to change.

This thesis describes the application of the relational database model for a labor, material, and job tracking system in a manufacturing setting. The tracking system is within a Computer Integrated Manufacturing (CIM) system, and a relational database manager called ORACLE is used for its implementation. The concept of CIM is relatively new and true applications are even newer. A rational for its use is given, and the uses of databases within CIM are highlighted. An overview of the actual CIM system is outlined with a listing of its hardware and software components, its overall environment, and its functions.

To provide the reader with a basis for understanding database concepts, an overview of databases is described. Various types of databases are compared, and issues relating to relational database implementation are discussed. ORACLE, the relational database management system (RDBMS) used in this research, is a leader in relational technology. The importance of ORACLE is outlined in support of its selection. Also, the major components of ORACLE's SQL query language are highlighted.

The process of developing the database including the data gathering, structured analysis, and database design are development tasks that are discussed. The actual tracking system application is given consideration and its performance is evaluated. A summary and the conclusions drawn are then given.

I.2 - Why CIM?

As increasing numbers of countries rapidly progress through their own industrial stages of development, world markets increase in volume and competition. Manufacturing in particular is an area where competition has become fierce while countries with various economic scenarios use their advantages in labor rates, government support and a host of other factors. As one of the leaders in industrialization, the United States has experienced serious losses in manufacturing. In the decade of the '60's, the U.S. lost 16% of its share of the world market and lost another 23% in the decade of the '70's [Ref 11, pp 59].

Increased competition is not the only reason for the United States' fading dominance. The emphasis on manufacturing decreases further as more and more graduates are drawn into service jobs. The cost of American labor, the highest of all countries, certainly has been an impediment to effective competition. Continually changing and not always helpful government policies are a factor. Management even shares the blame by using a lax approach for improving plants and financial justification schemes that place to much emphasis on the short run and ignore strategic issues. Each manufacturing sector contains its own problems, but taken as a whole, "...the United States' ability to compete in the world economy has eroded significantly during the past two decades." [Ref 12, pp 34]

Now that the state of American manufacturing is in trouble it is being given significant attention. Implementing new and better technologies in manufacturing is a necessary and crucial step in correcting the problem and moving towards the factory of the future. While the identification of the specific technologies to implement are arguable, there is strong agreement on the need for computer integration in the manufacturing processes. Benjamin, Rockart, Morton and Wyman profess this view in their Sloan Management Review article "Information

Technology: A Strategic Opportunity".

> "The revitalization of the manufacturing industry in the United States leans strongly on information technology. For companies to compete effectively internationally, several technologies must be utilized. These include computer aided design and manufacturing, automated factories and robotics, and new approaches to inventory flow."
>
> [Ref 13, pp 4]

There has been enough written about Computer Integrated Manufacturing to fill the average shop floor. This proliferation of information has portrayed CIM technologies and strategies as the savior of manufacturing in the United States. Unfortunately, there is no one accepted definition of CIM. Each industry and organization has its own goals and needs which affect its views and potential benefits from CIM. Listing all the possible benefits is unrealistic; however, Robert Kaplan in his Harvard Business Review article *"Must CIM be Justified by Faith Alone?"* identified the following six general areas of CIM benefits [Ref 5, pp 89-92]:

> **Inventory Savings** - *resulting from faster throughput, greater predictability and reduced waste, scrap and rework.*
>
> **Less Floor Space** - *resulting from fewer machines and less inventory on the shop floor.*

Higher Quality - *resulting from more uniform production, a decrease in defects and fewer inspections.*

Greater Flexibility - *resulting from economies of scope, multipurpose programmable machines and easily accommodating engineering changes.*

Shorter Throughput and Leadtime - *resulting from reduced WIP and better machine utilization.*

Increased Learning - *resulting from gaining experience with new technologies, testing the market for new products, and keeping a close watch on major process advances.*

With these benefits in mind, it is clear why so many corporations are implementing CIM technologies in their organizations.

I.3 - Databases in CIM

Information is perhaps the single most important resource in industry today. As more and more information service companies enter the market place and computing technology progresses, the strategic importance of information becomes more apparent. The concept of CIM is based on the goal of having all the information that relates to manufacturing decisions available when the decisions are made [Ref 2, pp 3]. Databases are one means of storing and organizing data in order to retrieve necessary information when it is needed for decision making.

Certainly with improved speed and organization of information provided by databases, many benefits will result. Information available when needed for decision making with help reduce leadtimes and inventory. Higher quality can be achieved with an increase in the amount and types of quality data that can now be stored and accessed easier. Flexibility will result from the various options allowed by querying the stored data for answers to new questions. As learning about this technology increases, new benefits will arise and applications will increase in number.

Databases are currently used for various purposes in manufacturing. The most common application functions are:

Computer Aided Design *to store the part geometry data.*

Material Requirements Planning *to store the bill of material data.*

Computer Aided Process Planning *to store the part's major features.*

These areas are currently benefitting from the application of databases; yet, there are so many other manufacturing areas in which these systems can prove useful. For instance, they can be used in quality control applications to store quality data or in tracking systems maintaining status of labor or material or in artificial intelligence systems that assist in manufacturing decisions and processes. Their possibilities are not boundless, but they are vast.

In their article, "Managing Multiple Databases in CIM", MacDow and Dirk of Prime Computer, Inc. state "There is practically no one in the database world who openly advocates multiple databases as a design goal. However, every organization in fact has many databases to deal with [Ref 2, pp 1]." They go on to explain there is a "company myth" of centralized database capability but it never reflects the existing situation. This myth has occurred because the same assumptions used for mass production have influenced database development:

- Bigger is better than smaller.
- Centralized is better than local.

o Speed is better than flexibility.

[Ref 2, pp 2]

With the emergence of CIM these assumptions will no longer be valid and the myth of completely centralized database capability will be shattered. The information needs of the various groups and tasks in an organization will multiply rapidly and the result will be:

1) End user demand for sharing unintegrated information will increase dramatically. Resulting in more distributed control.

2) Graphical and non-graphical information must be integrated.

3) Time pressures for sharing information within the CIM environment are greater than methodologies for data integration will allow.

[Ref 2, pp 1]

The Manufacturing Studies Board, Commission on Engineering and Technical Systems formed by the National Research Council has published similar findings in their book Toward a New Era in U.S. Manufacturing *The Need for a National Vision*:

"Current practice finds a large number of information retrieval systems in place even at the same company. Each system is associated with one major function, such as accounting, shop floor information, material requirements planning, or quality control statistics. ...experts predict that future manufacturing databases will be 20-50 times larger than present databases. The size of the databases,

> the time tolerances for communication, and the variety of users suggests that a manufacturing system database will be distributed across multiple heterogeneous systems, which may be in different geographical locations."
>
> [Ref 3, pp 109-111]

I.4 - Summary

CIM is an important part of revitalizing the competitive edge of U.S. manufacturing. Coupling information technology with manufacturing technology can help make this a reality and boost productivity. Databases are a particularly useful data processing tool to help organize, store and retrieve manufacturing data. The database systems that are in use in manufacturing today are generally dedicated to specific functions. Even though combining the individual databases is a preferred objective, the constraints and demands on the data make this an unlikely possibility for the foreseeable future. The relational model, the newest database system, has not been used much in manufacturing settings and has therefor been the main focus of this thesis.

Now that the basic reasons for implementing CIM and the importance of databases in the future of these systems are apparent, a description of the specific CIM system for this thesis is needed. With this described, the reader is provided a frame of reference. Chapter II outlines this CIM system, then chapter III describes these database components in greater detail.

II THE CIM SYSTEM

II.1 - Introduction

Before examining databases, the specific CIM system into which the database will be incorporated, must be discussed. CIM usually signifies a broad integration of manufacturing functions. While this specific CIM system is not all encompassing, it still represents a first step in elevating the use of computers for integration on this shop floor. Important information is given describing the physical environment for this system, the system functions, and the actual hardware and software.

II.2 - The Physical Environment

The lower end of the CIM system is located on a shop floor cell, known as a suspension cell, used for classical machining of large metal parts (more on the system's configuration will be given in a later section). The cell's layout is long and narrow covering approximately 30 ft by 180 ft (see figure 1 on the next page). It includes

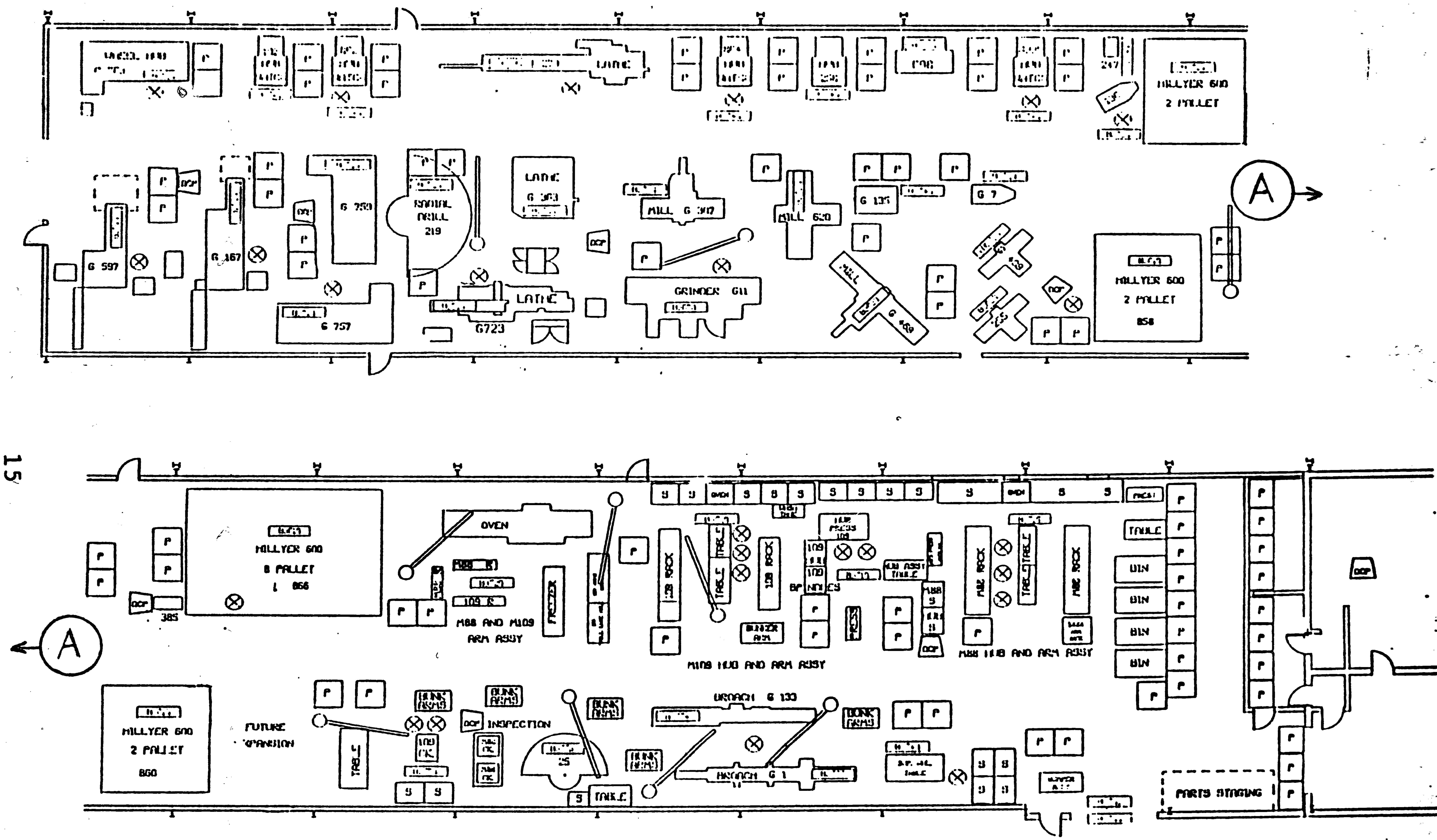


Figure 1 : Shop Floor Layout

a variety of machine tools such as lathes, milling machines, drill presses, grinders, and turning centers. The volume of parts is relatively low, only about 50 different part numbers move through the cell each week. Each part is machined in batches with typical lot sizes in the range of 40 to 60.

The information flows that exist are all manual (by paper and word of mouth), with no real provisions for determining status of material, job, or employee. When material is finished being machined in other machine cells, it is palletized and delivered to one end of the suspension cell. Attached to each pallet is a "picklist information tag" that contains the part number, workcenter number, work order number, quantity, and next higher assembly number. The foremen in the suspension cell must continually check what material has arrived in order to determine what jobs are ready to be released to the cell. Then, ready jobs are prioritized. The highest priority jobs have "job packets" assembled for them and are released to the first machine center on the part's routing. The job packet consists of the part drawings, the material count, and specific work instructions for that step which includes the next destination to deliver the parts when the operation is complete.

Once a job is released, the only means for determining its status is by physically checking it. The foremen must question each operator to determine the last job completed, where the material was placed, and what is the status of the current job.

Employee clock-in and clock-out is performed with time clocks and cards located at the entrances to the suspension cell. Only clocking into or out of the cell is recorded at these sites. Operators manually record time spent setting up and running jobs at their workcenters.

All CNC machines are run independently, each with its own controller and tape reader. Programs for these machines are written in another area and punched into tapes. Each tape must be walked down to the machine for testing. If errors are found, the program must be edited and a new tape punched off-site and brought to the machine for a new test.

II.3 - System Functions

Clearly, with the above description of the environment, one can find numerous areas for improvement through the use of computers. This CIM system addresses several problems in the cell. The main areas of concentration are to provide:

- Graphics capabilities.
- DNC capabilities.
- Labor tracking.
- Material tracking.
- Job tracking.

Graphics capabilities on the shop floor eliminates the need for distributing part prints in the job packets and therefore reduces the packet bulk. Requesting the drawing on-line when it is needed, instead of referring to prints that were created at some time in the past, eliminates the possibility of receiving incorrect versions.

DNC capabilities for all the CNC machines in the cell eliminates the need for paper tapes. Part programs are sent directly to the appropriate machines and are tested. When program mistakes are found they are edited locally at the nearby terminal.

The last three items are all related to tracking. These lend themselves well to manipulation by software, and they represent the center applications of the relational database system.

II.4 - Configuration

There are few turnkey solutions to CIM problems today; therefore, most CIM systems are a combination of various pieces of hardware and software. This CIM system is not unlike most in that it too contains hardware from IBM, DEC and Western Gear Corporation and software from McDonnel Douglas Corporation, Western Gear Corporation, Cullinet, and Oracle Corporation.

The majority of the system's hardware will be physically located near the workcenters on the shop floor. Each CNC machine tool has one dedicated Western Gear Corp. Industrial Data Station (IDS). This IDS is basically an industrial hardened PC with an IBM XT architecture. Two more IDS's are in the shop, one at the

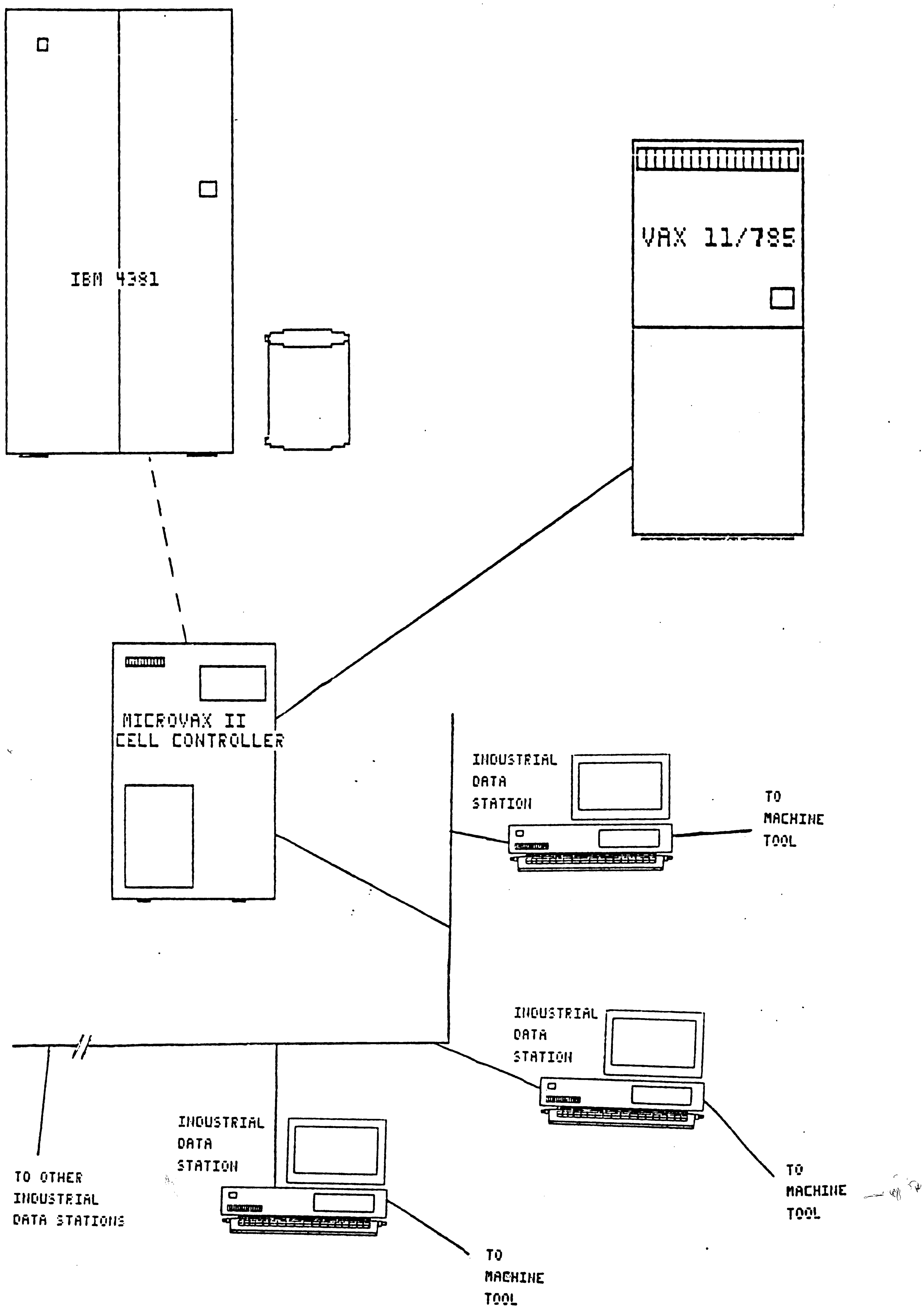


Figure 2 : CIM System Configuration

far end to record material that is received in the cell and the second at the other end in the foremen's office. All these PC's will tie into the cell controller, a DEC MicroVAX II (see figure 2 on the next page).

The heart of the system, the DEC MicroVAX II, will be physically located in a remote computer room. This cell controller will have two other main links. One link will be to a VAX 11/785 and the other link will be to an an IBM 4381. Both the VAX 11/785 and the IBM 4381 are located in separate, remote computer rooms.

The IDS for each machine tool will perform a variety of functions:

1) Act as machine controller to its dedicated machine tool. The IDS runs Western Gear's DNC software that enables the operators to transfer completed programs down to the local hard disk. Once a program is local, it can be tested on the machine through a dry run. Program mistakes can be edited locally and new tests performed right away.

2) Provide graphics support of the part prints to the operators. The IDS emulates a graphics terminal and displays part drawings stored on the MicroVAX II.

3) Provide tracking information support to the operators. The IDS interacts with the MicroVAX II cell controller and performs transactions with the ORACLE database manager running there.

The MicroVAX II acts as the suspension cell controller. It is the focal point of the system since it is the main repository of the following data ready for access and distribution:

Part programs created on the VAX 11/785 and waiting for downloading to the individual IDS's.

Part drawings created on the VAX 11/785 and waiting to be viewed by the operators working on related jobs on the shop floor.

Tracking information created on the MicroVAX II and manipulated through all the IDS's.

The MicroVAX II also runs the following software:

- o DNC software which maintains and tracks all part programs for each IDS.
- o Unigraphics II CAD software which maintains and tracks all part drawings.
- o ORACLE relational database manager which maintains and organizes all the cell tracking information.

The VAX 11/785 resides in an engineering department and supports two functions:

- o Remote creation of part prints with CAD software.
- o Remote creation of part programs.

The IBM 4381 is a planned future expansion that will interface the ORACLE database tracking system with a Cullinet scheduling system.

II.5 - Summary

Much of the CIM system is actually on a metal machining shop floor interfacing with the machine tools, the operators and the foremen. The system reduces the need for paper, increases productivity and provides easier and quicker access to shop information by providing graphics capabilities, DNC capabilities, and labor, material and job tracking. These functions are all supplied through an integration of various vendor's hardware and software.

Now that the pertinent system facts have been discoursed, the database's required functions and the framework it must operate in are more distinct. Next, chapter III describes databases manifesting their importance and characteristics. Chapter IV describes relational databases in greater detail and chapter five highlights the specific relational database manager chosen, ORACLE.

III DATABASES

III.1 - Introduction

Now that the CIM system is understood, the major component called the database must be discussed. Database systems are defined, and reasoning for their development as data processing tools are given. Their advantages and disadvantages are discussed, and the three major types of databases are outlined.

III.2 - Database Systems

C. J. Date, currently one of the most noted experts on databases, provides the following definition for database systems:

> "A database system is essentially nothing more than a computerized record-keeping system. The database itself can be regarded as a kind of electronic filing cabinet -- that is, as a repository for a collection of computerized data files. The user of the system will be given facilities to perform a variety of operations on such files...."
> [Ref 1, pp 3]

There are six main functions that the database system must perform for the users. They are:

- o Add new files to the database
- o Insert data into files in the database
- o Retrieve data from files in the database
- o Update data in files in the database
- o Delete data from files in the database
- o Remove files from the database

These represent the core functions that allow for effective utilization of the database system.

III.2.1 - Components of a Database

One view of a database system contains the four components: data, hardware, software and users. Figure 3 below provides a greatly simplified view of these four elements [Ref 1, pp 6].

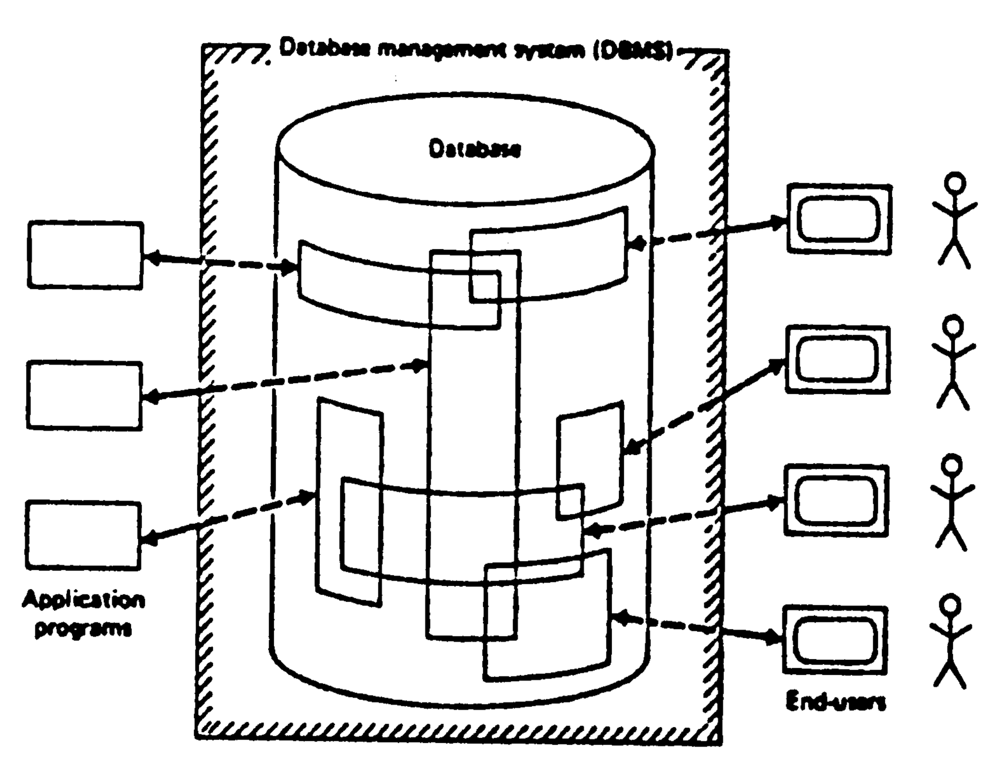


Figure 3 : A Simplified View of a Database System

Data

Data stored in a system is partitioned into one or more databases. Bits of data are grouped into bytes or characters, characters are grouped into fields, fields are grouped into records and records are grouped into files. Files in a database are distinctly different from those in traditional file processing systems. Databases contain a collection of "integrated" files with relationships among records and fields in those files (the next section will elaborate on file and database system differences).

Hardware

The vast majority of database systems do not require special hardware for their operation. The typical computer system consisting of the generic components 1) Central Processing Unit (CPU), 2) Direct Access Storage Devices (DASD) or secondary storage and 3) a Terminal Interface are sufficient for most database systems. There are several vendors that provide database machines for performing database processing functions. Requests for database services are routed to the database machine by the CPU. This allows for database processing to continue without degrading the responses associated with the CPU processing. Aspects of hardware requirements are a separate topic that has already been sufficiently documented and will not be considered further in this thesis.

Software

The database management system (DBMS) is a software layer that provides an interface between the users and the physical database (stored data). All user requests for database functions are handled by the DBMS. This separates the users from the hardware-specific and low-level details. The separation provides an elevated view and supports operations that are expressed in terms of this higher level view. More will be discussed concerning this separation in a later section.

Two other software components predominant in database processing are the operating system and application programs (see figure 4 below).

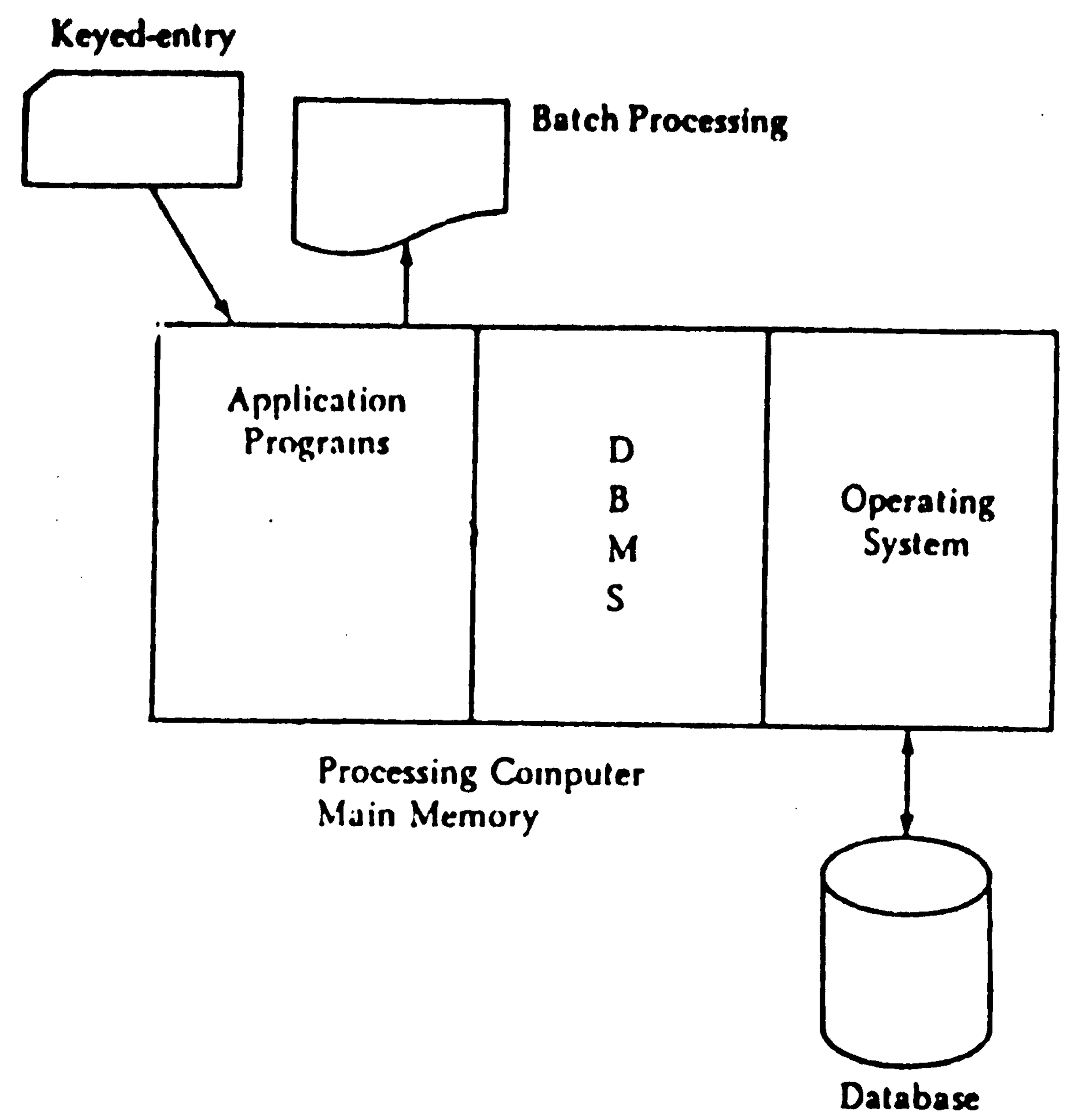


Figure 4 : Software Components of a Database System [mostly from Ref 4]

Application programs are simply programs written to perform some specific function. For example, an application program can be written to generate a report listing the status of all jobs in progress. Application programs that need access to the database must request service from the DBMS.

The operating system is a basic part of any computer system. It controls the system's resources including the secondary storage devices where the database normally resides. All database requests must be routed through the DBMS but the DBMS must interface with the operating system in order to provide the machine level information needed to fulfill each request. Operating systems are specific to the computer system used and are outside of the database system; therefore, they will not be expounded on further.

Users

There are three classes of database users 1) application programmers, 2) end-users and 3) the database administrator.

Application programmers are responsible for writing the application programs that use the database. These programs are usually written in a high level language such as COBOL or FORTRAN. They may be batch applications or on-line programs supporting end-users on the system.

End-users can access the database through a terminal. These users can interface with the database by using pre-written application programs or by writing their own ad hoc queries.

The database administrator (DBA) is responsible for overall control of the database system. The main responsibilities are to protect the integrity of the database and to resolve users' conflicts.

III.2.2 - Three Levels of Architecture

A database system is divided into three logical views or levels (see figure 5 below) [Ref 1, pp 29-30].

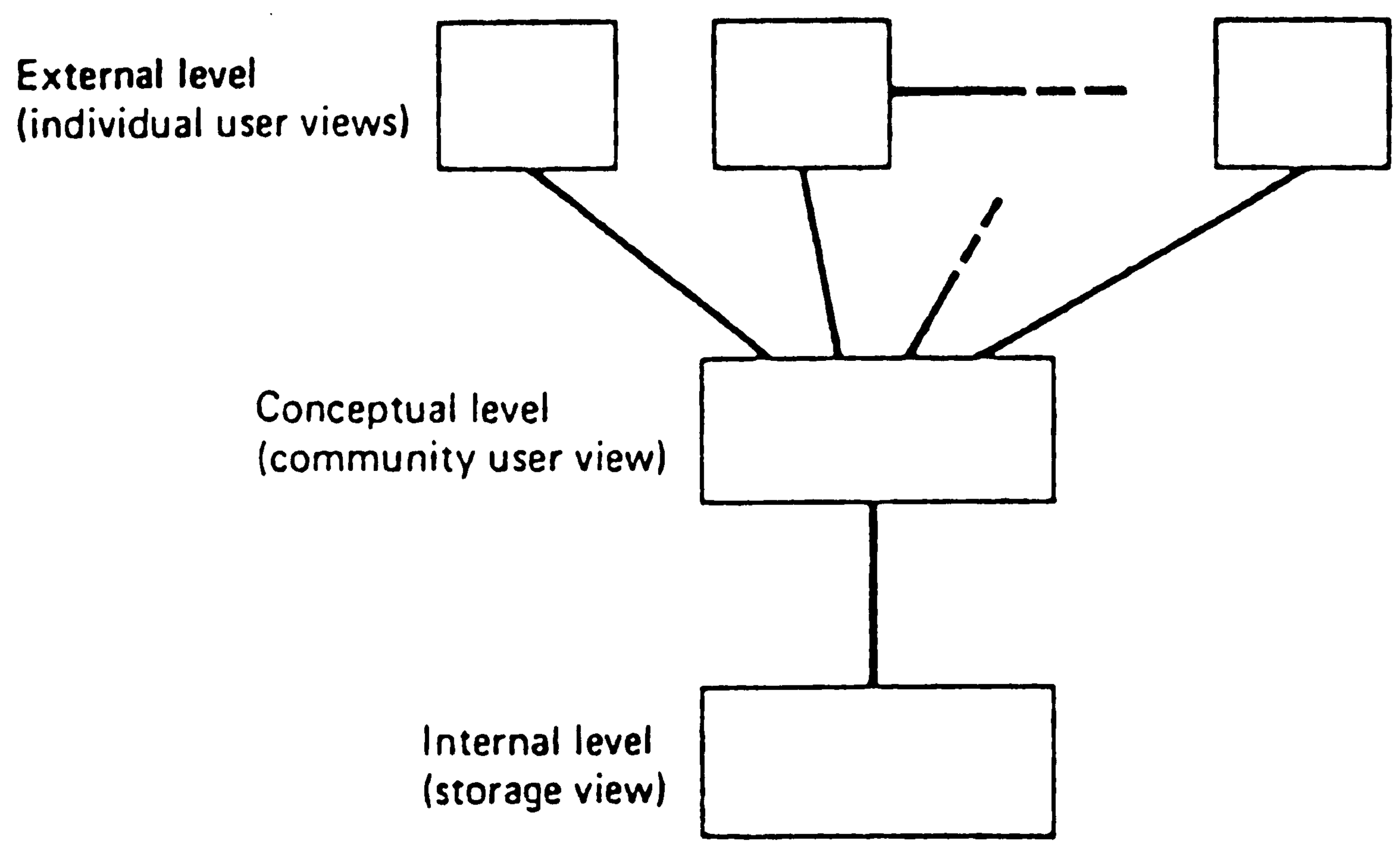


Figure 5 : The Three Levels of Database Architecture

These levels are:

Internal Level - The level closest to physical storage. It is concerned with the way data is actually stored.

External Level - The level closest to the users. It is concerned with the way the data is viewed by individual users.

Conceptual Level - The level of "indirection" between the other two. It is concerned with the way data is viewed by the whole community of users. It takes into account all of the individual user views.

III.2.3 - Operational Data and Information

Data stored in a database is often referred to as "operational data" distinguishing it from input, output and other types of data. More specifically, operational data is data necessary for decisions and the operations of some group or organization. Therefore, a functional definition of a database is a collection of the stored operational data used by the applications of a particular organization.

Information and data have distinct meanings. Data is unprocessed facts that reside in the database (the operational data). Information is the derived meanings from the interpretations of data. Data are processed into information in order to be employed by the users.

In certain cases, data can be used as information without further processing. In general, however, data is not useful until processed and presented in an organized form. Figure 6 below displays an example of data processed into information.

5 pcs
$100
$40
MMQ Inc.
6 pcs
2 pcs
$120

Data Processing →

Customer: MMQ Inc.
January 1987 Monthly
Purchase Summary

Total ==> $260.00

Data

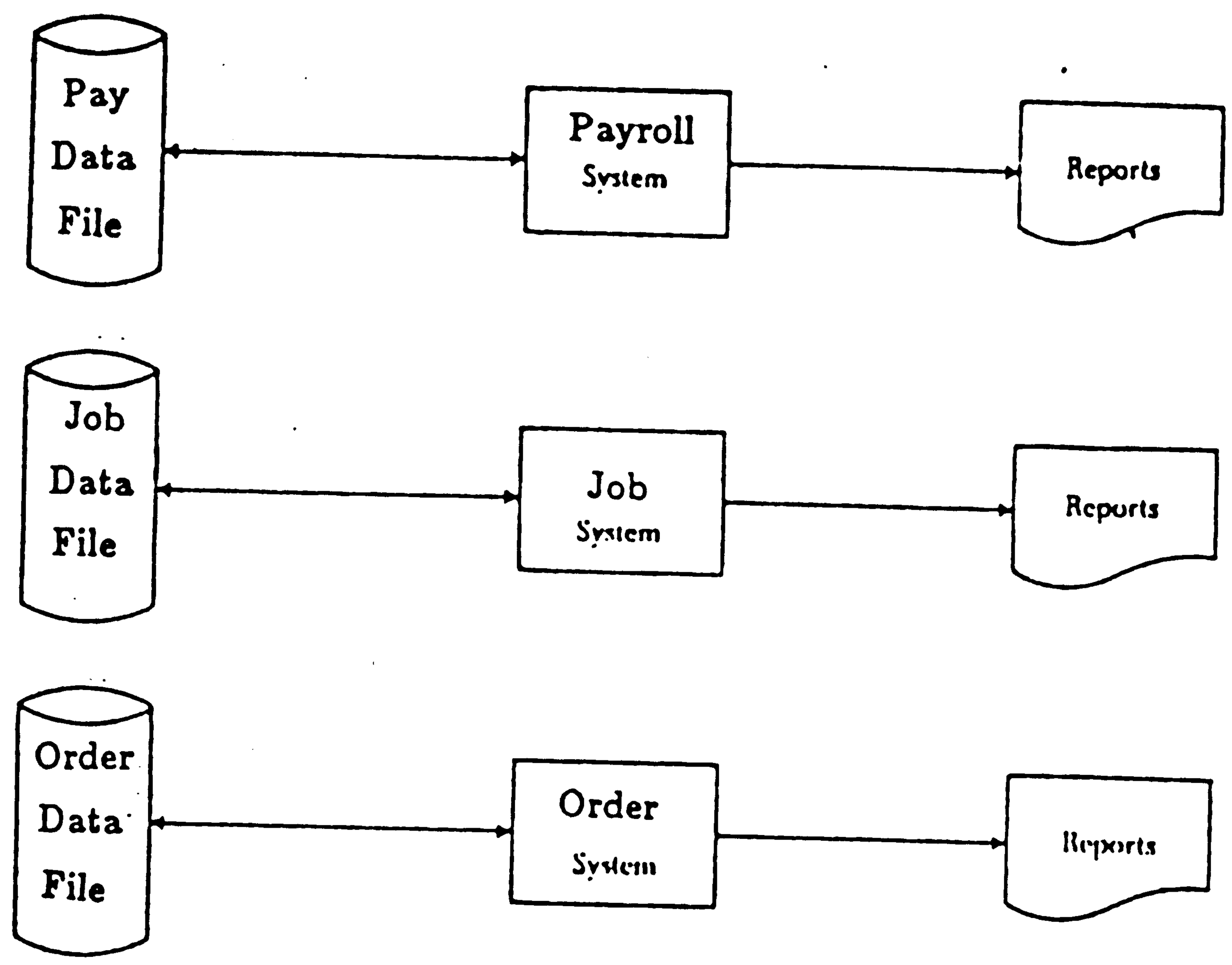
Information

Figure 6 : Data Processed into Information

Hence, the operational data stored in the database is of little value until it has been processed by applications into meaningful information.

III.3 - Why Database?

Organizations that use database technology are better able to process data as an integrated whole. Data processing systems that existed before databases where composed of separate files for separate applications. These individual files imposed an artificial division when accessing data. For example, data could be stored in three different files specific to certain functions. One file could contain employee payroll information, a second file could contain job status information, and a third file could contain order information (see figure 7 below).



[mostly from Ref 4, pp 2]

Figure 7 : Three File Processing Systems

The payroll system only processes hour and rate data, the job system only processes job status data, and the order system only processes order data.

Suppose someone requested job costing information using employee rate data and job data, or someone requested the status of all jobs for all orders from a particular customer. To satisfy these requests, new programs must be written which pull data from each of the files it needs. This can usually be done, however, it is frequently costly and time consuming. Sometimes the files cannot be used together by the same program because they are in different formats and hence, incompatible.

III.3.1 - **Advantages of Databases**

The one main advantage for using a database system is to provide the enterprise with centralized control of its operational data. As just shown, most enterprises have individual files that pertain to each of its applications causing the operational data to be dispersed and hard to control. For a database system, the DBA oversees and coordinates this central control.

Central control spawns a host of other advantages. They are:

- o Redundancy can be reduced
- o Inconsistency can be avoided
- o The data can be shared
- o Standards can be enforced
- o Security restrictions can be applied
- o Integrity can be maintained

o Conflicting requirements can be balanced

[Ref 1, pp 13-15]

An important advantage of database processing is the elimination or reduction of data duplication. When private files exist for each application, there is often redundant storage of data. Storing data only once in the database reduces wasted space and can result in reducing processing requirements. There may exist some business situations where redundant data is needed but at least the database allows for the redundancy to be controlled. Data duplication can produce data inconsistency and a lack of data integrity.

Sharing data enables more information to be produced from the same amount of data. Also, it may be possible to satisfy the data requirements of new applications without having to create additional stored data.

The DBA can enforce all data standards to maintain data that is easy to understand. This is particularly useful when data is being exported or imported, or during migration to a new system. Since the DBA has central control he can implement security measures that ensure the only access to the database is through specified channels and that sensitive data can be restricted from selected users.

Data integrity refers to the accuracy of the stored data. This integrity can be affected by redundancy and incorrect insertions or updates. Integrity checks can be placed to eliminate the possibility of incorrect data.

Since the DBA knows the overall requirements of the enterprise (collective requirements) he can structure the system to provide the best overall balanced service.

III.3.2 - Disadvantages of Databases

Like most technologies, using databases has its advantages and disadvantages. Some of the disadvantages of databases are:

- o Cost
- o Complexity
- o Vulnerability

[Ref 4, pp 6-7]

Database systems can be expensive both to purchase and to use. Initial costs for these systems vary widely, generally ranging from $50,000 to $100,000 for systems on mini and mainframe computers. Adding training and vendor support make the start-up costs formidable. Besides high start-up costs, the database system has a tendency to consume large amounts of computer resources. So much main memory may be occupied that additional amounts will be needed. Memory increases may not sufficiently satisfy the database's needs if it monopolizes the CPU. A larger computer system is then required which can lead to considerable additional expense for the purchase and conversion.

Some types of processing are not appropriate for using a database and can cost much more if a database is used. For example, sequential processing of records, such as in a payroll system, is particularly well suited to file processing systems. Using the database for this type of application will produce results that are slower and more costly due to the increased overhead of the DBMS.

The increased organization and flexibility provided by databases stem from its complex structure and processing. Data stored in the database is usually stored in various formats and has interrelationships. Because of these characteristics, the database manager and application programs must be more sophisticated. This implies that creation of applications will take longer and require higher programmer skills. These characteristics also require more complex backup and recovery procedures since it compounds the determination of the exact state of the database at time of failure.

A last disadvantage comes from the advantage of a database's central tendency. While the centralization provides central control and all the advantages listed in the last section, it leaves the enterprise vulnerable to system failure. If indeed there is only one database located in one computer system, failure of the hardware or database software will suspend access to the users. Consideration should be given for alternate or parallel support of functions that are critical or that depend heavily on the database. One option is to maintain a manual system for use during system downtimes. Another option is to design in a type of redundancy where the database is distributed over several machines. If one machine fails, an alternate machine can take over the disabled machine's functions while repairs are performed.

Even though disadvantages with databases exist, they are not great enough to prevent their use. As this technology progresses, the purchase and operating costs decrease, the DBMS handles more and more of the complex issues including backup and recovery procedures, and strategies are evolving to reduce the risk involved with vulnerability. With the continuation of these trends, database implementation will continue to increase.

III.4 - Types of Databases

Modeling of data is the fundamental concept involved in the development of database systems. Different models provide different operating characteristics. Many different models have been created, but the three most prominent are hierarchical, network and relational models.

All three of these classical models use the record model for their basis. In the record model, data is arranged into fixed linear sequences of field values. This record setup is machine oriented because it is an efficient means for storing and processing data. Unfortunately, this machine orientation is not also user oriented. The different models of database managers provide user orientation with an interface consisting of a set of operations associated with the model structure.

Hierarchical data model

Data in the hierarchical model is arranged into tree structures. Therefore, a hierarchical database is merely a collection of trees. This organization requires that some records are subordinate to others meaning that parent-child relationships exist. The tree structure provides meaning for each record; records out of the structure are out of context. Figure 8 below provides an example of the structure in a hierarchical database.

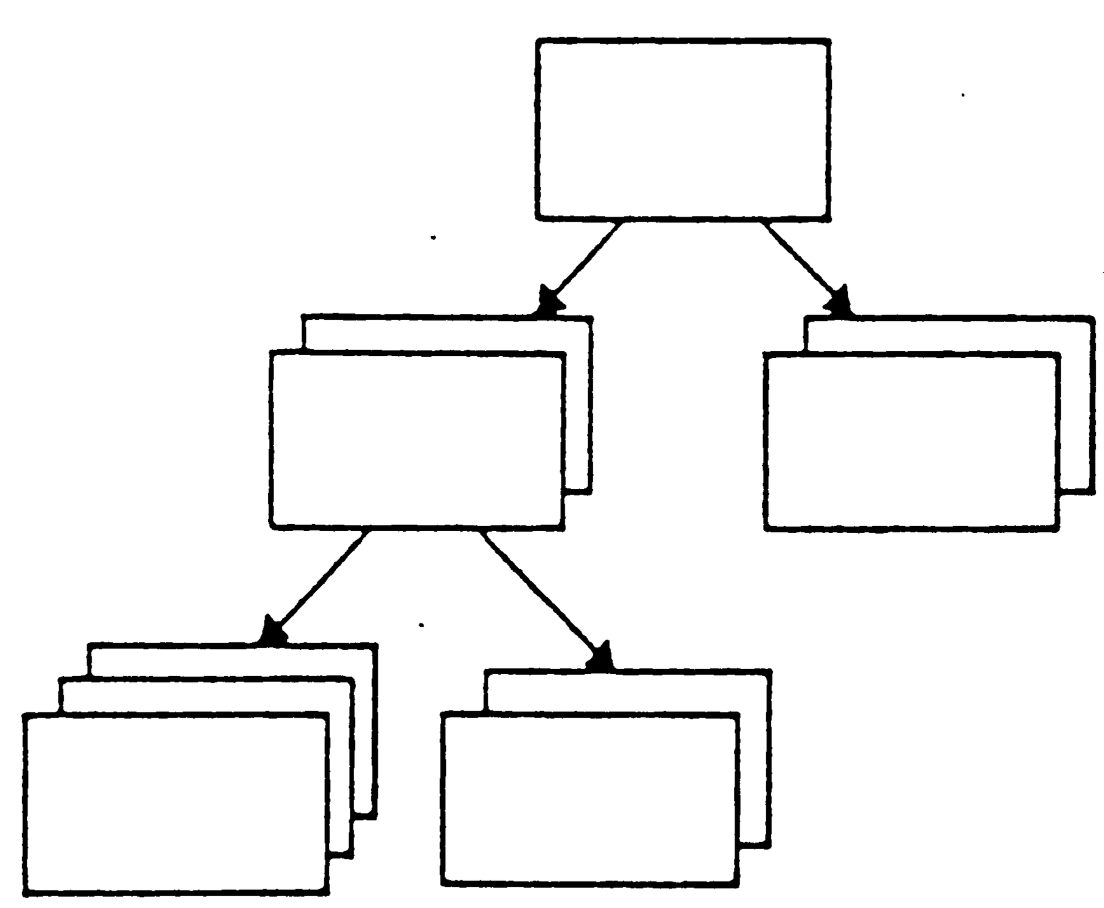


Figure 8 : Hierarchical Database Model Structure

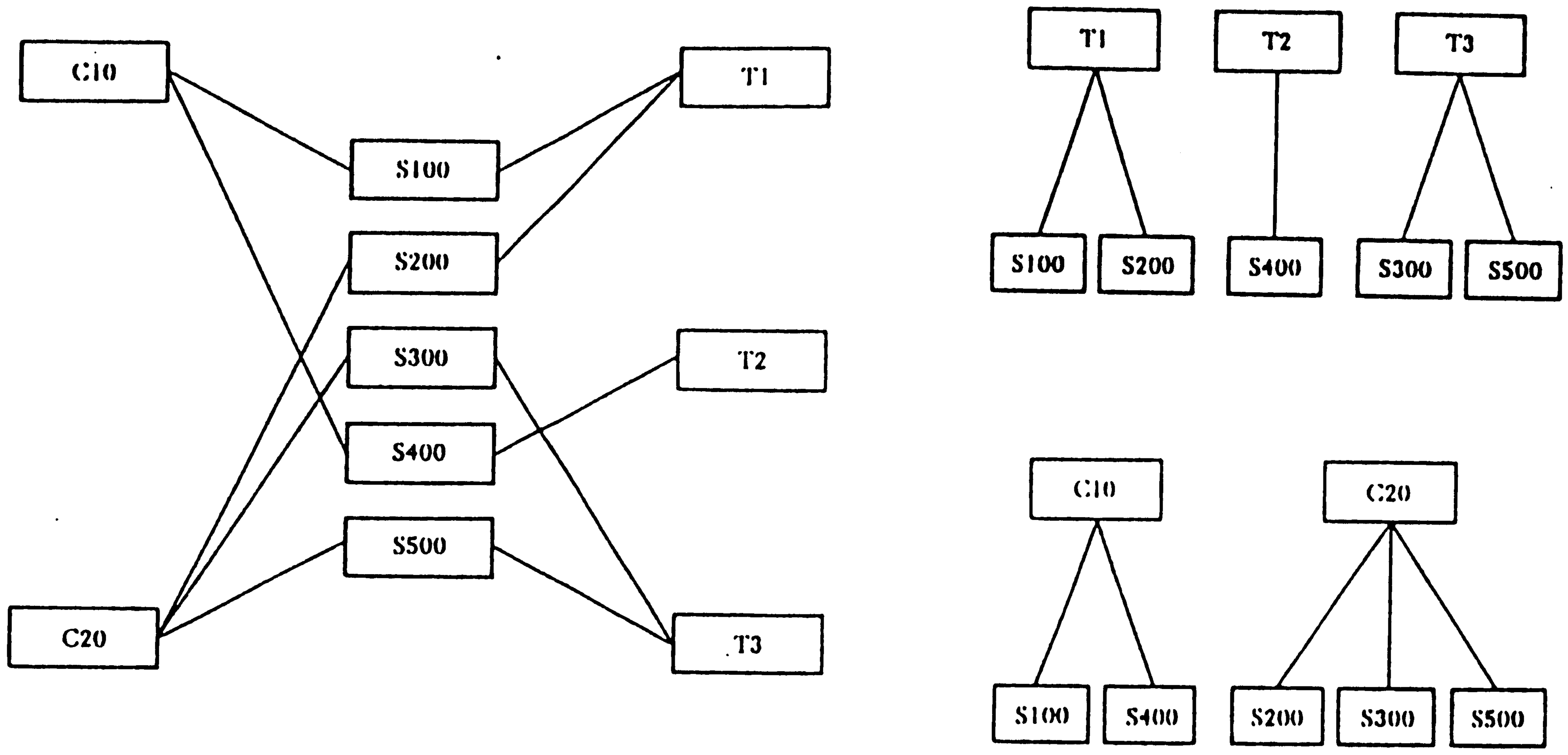
A true hierarchical structure, however, is not a natural model for many database applications. Because the hierarchical structure is limited, the virtual logic record type has been invented which points to another record type. These pointers provide a mechanism that allows a record to exist in several trees. Without this facility, records would have to be duplicated resulting in data redundancy.

Operations for a hierarchical data model are inherently very low-level and procedure oriented. This requires the users to understand the internal data organization and the predefined data relationships.

There are numerous hierarchical database management systems commercially available today. The most popular of these is IBM's Information Management System (IMS).

Network data model

Data in the network model is represented by records, interconnected by links to form what is called a directed graph. To accommodate a many-to-many relationship, records are grouped into sets, each of which consists of one owner record and zero or more member records. For implementation in the database this requires a duplication of records to bridge the two sets (see figure 9 below).



logical network   decomposed network with duplicates
[Ref 4, pp 140-141]

Figure 9 : Network Database Model Structure

The duplicate record, sometime referred to as a connection record, contains the information common to the records linked by the many-to-many relationship.

In general, the network data model is more symmetrical than the hierarchical data model. This connotes that the operations on this model are more symmetrical than operations on the hierarchical model, however, the network model operations are also inherently procedural requiring the users to be familiar with a low level of data representation.

There are a few commercially available network database management system today. Cullinet's Integrated Data Management System (IDMS) product is by far the most widely recognized of these systems.

Relational data model

The relational data model is the model used for this research, and it is discussed in depth in the next chapter. It is only given a cursory consideration here.

Data in the relational model is organized into tables. Each table (or relation) basically has a matrix layout where the columns are called attributes (or fields) and the rows are called tuples (or records). Figure 10 below displays the typical relational table format.

EMPLOYEE RELATION

EMPLOYEE NO.	NAME	SALARY
010	B. Smith	18,000
020	J. Jones	28,000
034	M. Zedway	39,000
042	F. Ben	78,000

Figure 10 : Relational Database Model Structure

A relation name with its set of attribute names is called a relational schema (or a relation), and a collection of relational schemas is a relational database.

The main attraction of the relational model is its mathematical clarity. It facilitates the formulation of nonprocedural, high-level queries which separates the users from the lower level data organization. Because of this facet, the relational data model is considered the most user oriented of the three classical database models.

There are several relational database management systems available commercially today. These include IBM's DB2 and SQL/DS. Oracle Corporation's RDBMS, ORACLE, is the most popular of these relational managers today. This manager was used for this thesis and chapter V provides information on its characteristics.

III.5 - Summary

A database system is simply a computerized organization system that stores operational data and is composed of four physical components: data, hardware, software, and users. These four components create three logical levels of architecture which are internal, external, and conceptual. Databases provide many advantages over their file system counterparts, but their main advantage is central control of the operational data. Disadvantages of cost, complexity, and vulnerability do exist but are not sufficient to deter their use. Of the three main types of database systems, the relational model is considered to be the most user oriented since it separates the users from the data implementation to the highest degree.

Now that databases have been described in general, the particular model used for this thesis, the relational model, will be focused on next. The ORACLE database manager, the specific database manager used, will then be discussed followed by information on the development of the database.

IV RELATIONAL DATABASES

IV.1 - Introduction

With basic database concepts now explained, the relational model with some of its properties must be discussed. This chapter provides information on the pertinent relational concepts of keys, relational algebra, modification anomolies, data duplication and normalization.

IV.2 - Relational Properties

The relational model was first introduced to the data processing community in 1970 by Dr. E. F. Codd. This innovation stressed the independence of the relational representation from the physical computer implementation such as ordering on physical devices, indexing, and using physical access paths. Thus, it was the first model to formalize the separation of the user's view of the data from its implementation. Codd also proposed criteria for logically structuring relational databases and an implementation-independent language to operate on these databases.

As previously mentioned, the relational model stores data in table form. This is attractive because there are formal criteria called normal forms that aid in setting up the table structure (normal forms will be discussed in a later subsection). The two dimensional tables called relations have several properties. First, the entries in the table are single-valued; neither repeating groups nor arrays are allowed, hence, the relation has a flat file format. Each tuple (row) is referred to as an N-tuple, where N is the number of attributes (columns) in that relation. The relation is then said to be of degree N.

IV.2.1 - Keys

The key is the attribute or set of attributes that uniquely identifies a tuple in a relation. A relation's key is formally defined as a set of one or more relation attributes concatenated so that the following three properties hold for all time and for any instance of the relation:

1) **Uniqueness** : The set of attributes takes on a unique value in the relation for each tuple.

2) **Nonredundancy** : If an attribute is removed from the set of attributes, the remaining attributes do not posses the uniqueness property.

3) **Validity** : No attribute value in the key may be null.

When there are more than one attribute or collection of attributes that meet these criteria, they are referred to as candidate keys. One of the candidates can be randomly selected to be the relation's key, it is then referred to as the primary key, and the remaining candidates are referred to as alternate keys.

Attributes in one relation that are keys in another relation are referred to as foreign keys. These foreign keys become important when defining constraints across relations.

IV.2.2 - Relational Algebra

Relational algebra is a collection of operations which may be performed on relations and is the mathematical basis on which Codd based relational databases. Each operation uses one or more relations as its operands and produces another relation as its result. For example, X = Y + Z can be interpreted to mean that the union (+) of operands Y and Z produces the resultant (relation) X.

Relational algebra consists of two main groups of operators; 1) set operators and 2) special relational operators. These operators are the core components of relational languages such as Structured Query Language (this query language is further described in a later section).

1) Set operators

The traditional set operators are union, difference, intersection, and cartesian product. The two relations used as operands must be union-compatible for all of these except cartesian product. This means that each relation must be of the same degree, and the attributes in corresponding columns must be from the same domain.

union

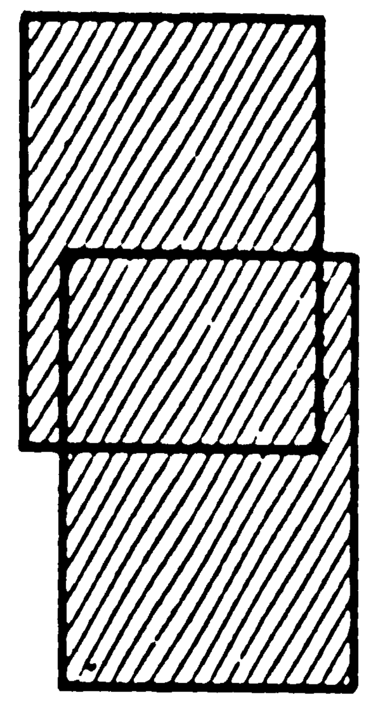


Figure 11 : The UNION Operation

[Ref 1, pp 259]

The union of two relations is formed by combining the tuples from one relation with those of a second relation to produce a third relation. Mathematically, the union of two relations, Y UNION Z, is the set of all tuples belonging to either Y or Z or the intersection of both. Duplicate tuples are eliminated.

difference

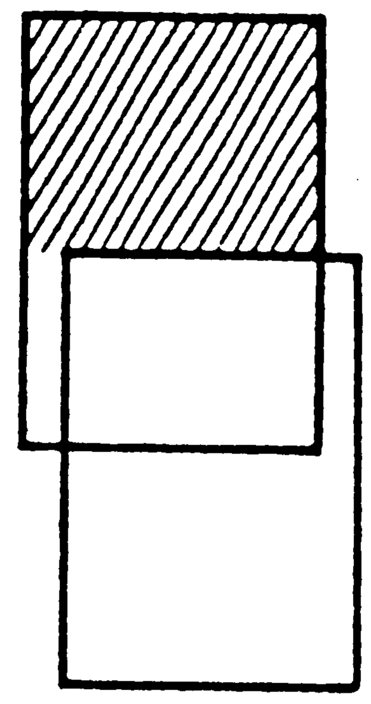


Figure 12 : The DIFFERENCE Operation

[Ref 1, pp 259]

The difference of two relations is a third relation containing tuples which occur in the first relation but not in the second relation. Mathematically, the difference between two relations, A MINUS B, is the set of all tuples belonging to A and not B.

intersection

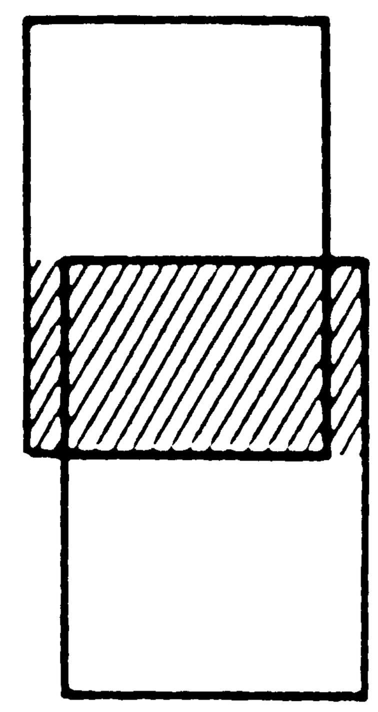


Figure 13 : The INTERSECTION Operation

[Ref 1, pp 259]

The intersection of two relations is a third relation containing common tuples. Mathematically, the intersection of two relations, A INTERSECT B, is the set of all tuples belonging to both A and B.

cartesian product

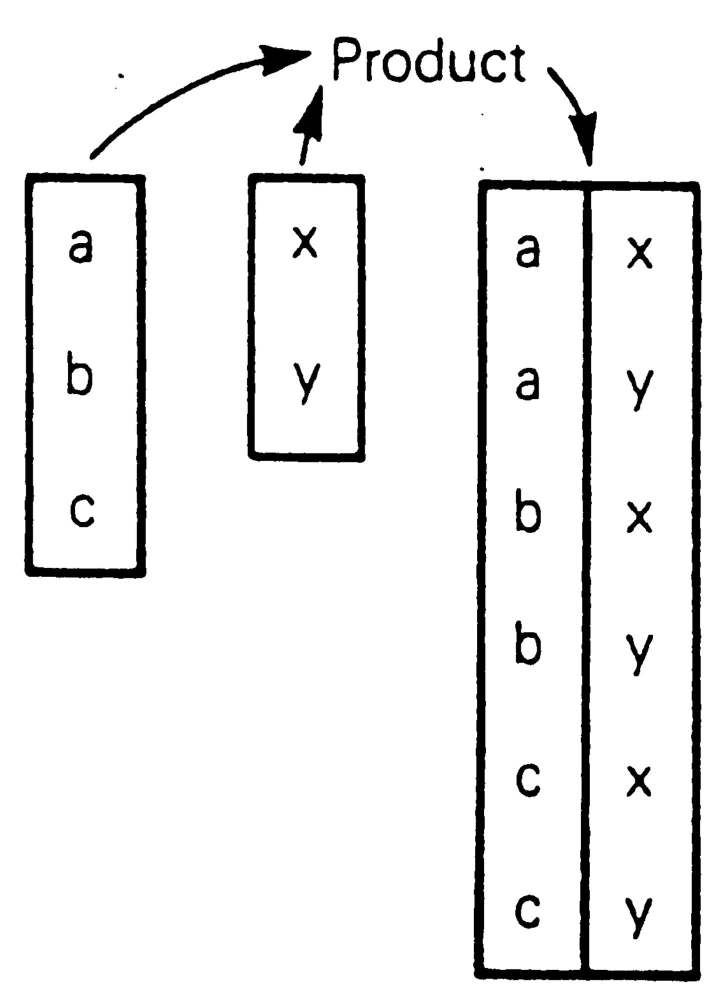


Figure 14 : The PRODUCT Operation

[Ref 1, pp 259]

The cartesian product of two relations is the concatenation of every tuple of one relation with every tuple of a second relation. Mathematically, the cartesian product of two relations, A TIMES B, is the set of all tuples $SUM_i(a_i \times SUM_y b_y)$, where a and b are the tuples for relations A and B respectively.

2) **Special relational operators**

projection

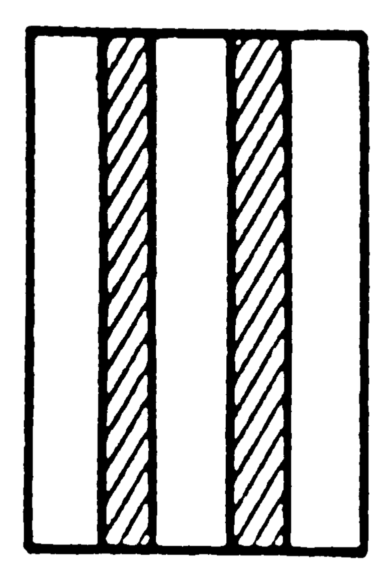


Figure 15 : The PROJECTION Operation

[Ref 1, pp 259]

Projection is an operation that selects specified attributes from a chosen relation. The result of the projection is a new relation having the selected attributes. This means a projection creates a vertical subset of a chosen relation by selecting certain attributes in a specified order. Duplicate tuples within the resulting relation are eliminated.

selection

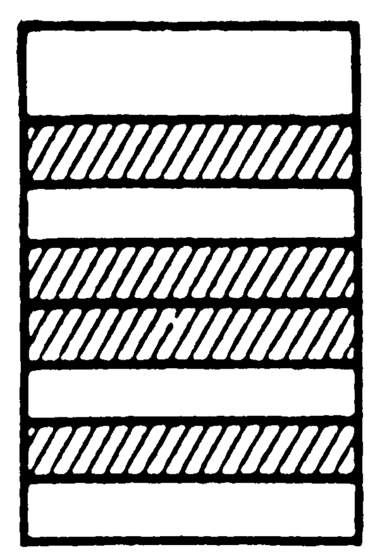


Figure 16 : The SELECTION Operation

[Ref 1, pp 259]

The selection operator produces a horizontal subset from a chosen relation. This means a selection identifies the tuples to be included in the resultant relation based on some selection criteria applied to the attributes.

join

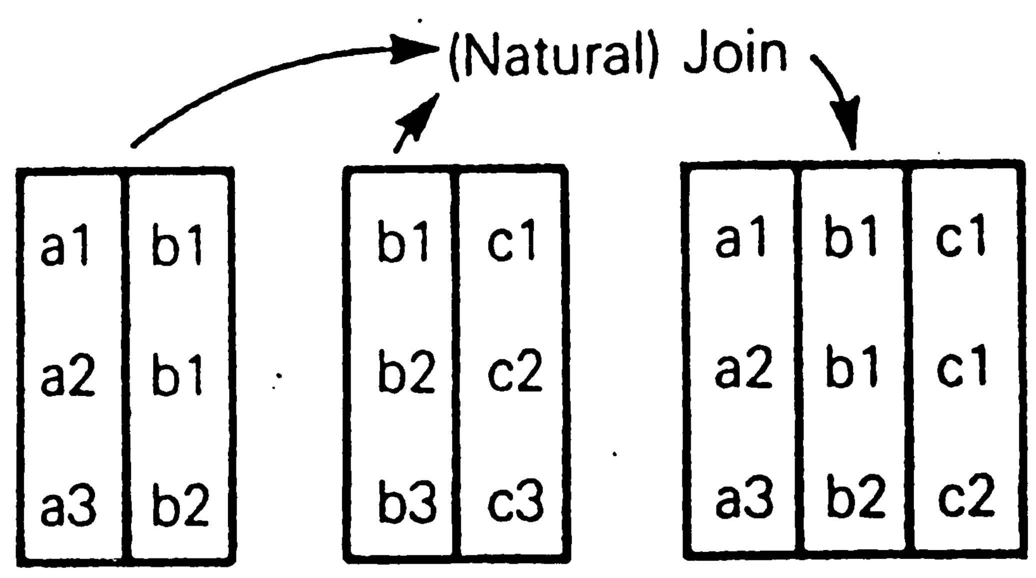


Figure 17 : The JOIN Operation

[Ref 1, pp 259]

The join operation is a combination of the product, selection, and sometimes the projection operation. The join of two relations, A JOIN B, is equivalent to taking the cartesian product of A and B and then performing a selection on that product. The join operation is considered a binary operation because it operates on two relations, while selection and projection only operate on one relation.

division

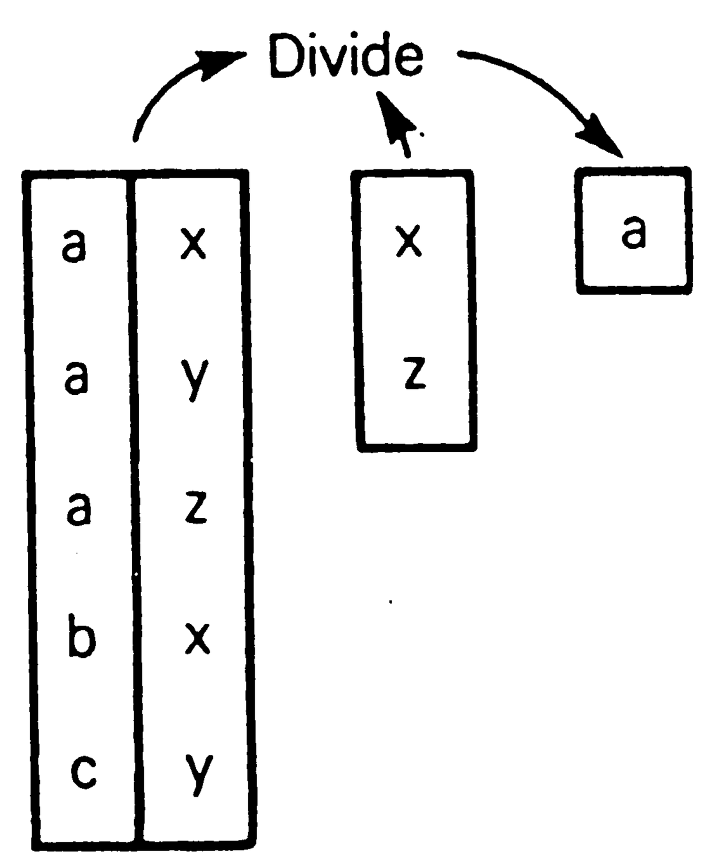


Figure 18 : The DIVISION Operation

[Ref 1, pp 259]

The division operation produces a quotient relation C of degree m by dividing a dividend relation A of degree m + n by a divisor relation B of degree n. The result is a set of tuples that contain the m attributes from relation A that shared the same n attributes with relation B.

IV.2.3 - Modification Anomolies and Data Duplication

Each relational database design has its own interrelationships. Changing data in some relations can have unexpected consequences that are undesirable called modification anomolies. Anomolies can be eliminated by changing the database design. This is usually preferred because unexpected changes to the data will not result; however, there are design situations where this is not feasible. Some relations are independent while others are interdependent; the less interdependent the less chance modification anomolies will exist.

The two major types of modification anomolies are insertion and deletion anomolies. An insertion anomoly occurs when inserting a fact about one data entity requires the insertion of a fact for another data entity. A deletion anomoly occurs when deleting a fact about one data entity causes the deletion of a fact about another data entity. Figures 19 and 20 illustrate these two anomolies.

Employee-job-location Relation

EMPNO	JOB	LOCATION
1234	Assembly	99-102
5466	Test	63-996
7772	Programming	23-942
5824	Assembly	99-102

Figure 19 : Modification Anomolies in a Relation

Figure 19 shows a relation that collects information on employees, their jobs, and the job locations. An insertion anomoly problem will occur if a new job, for example "design", is added to the list of jobs. This should be added to the database; however, it cannot be added until a valid employee number is assigned to this job and a valid location is assigned. A fact must be added about another data entity in order to add the original fact to the database; usually this is an unacceptable situation.

Figure 19 can also be used to explain the deletion anomoly. If the employee with number 7772 leaves the firm and his tuple is deleted, two important facts are lost. First, the fact that the employee was a programmer is lost. This is most likely an acceptable loss. Second, the fact that programming is performed at location 23-942 is also lost. This is an unacceptable loss resulting from poor database design.

Employee-job Relation

EMPNO	JOB
1234	Assembly
5466	Test
7772	Programming
5824	Assembly

Job-location Relation

JOB	LOCATION
Assembly	99-102
Test	63-996
Programming	23-942

Figure 20 : Modification Anomolies Removed

The relation design that eliminates the modification anomolies is shown in figure 20. A solution for storing data about employees and jobs independently is to create two separate relations. If employee 5824 is deleted, the fact that assembly is performed at location 99-102 is preserved. If employee 5466's work location is desired, two look-ups are required, one in each table. Further, if "design" is to be added as a valid job, an employee number is not required before it can be inserted.

Data duplication, another problem that can occur in database design, is illustrated in figure 21 below. This table contains information about an employee and his department. Much space will be wasted if this design is used because the same information is stored about the department for each employee in the department. The duplication occurs when there is information about two distinctly different data entities (EMPNO and DEPTNO) stored in the same relation.

Employee-department Relation

EMPNO	DEPTNO	DEPTNAME	DEPTLOC	DEPTCLASS
1111	10	ENGINEERING	99	G
2222	20	PURCHASING	88	Q
3333	10	ENGINEERING	99	G
4444	10	ENGINEERING	99	G

Figure 21 : Data Duplication in Relation

Eliminating data duplication is similar to eliminating modification anomolies. Two tables are created. One table for each data entity to store its related data items. Figure 22 shows the two tables necessary to eliminate the duplication in figure 21. This two table set up contains the DEPTNO in both tables to be used as a cross reference. Eliminating the duplicate copies of data in a database will significantly improve the database performance.

Emp-dept Relation

EMPNO	DEPTNO
1111	10
2222	20
3333	10
4444	20

Department Relation

DEPTNO	DEPTNAME	DEPTLOC	DEPTCLASS
10	ENGINEERING	99	G
20	PURCHASING	88	Q

Figure 22 : Data Duplication Removed

IV.2.4 - Normalization

There are many different types of modification anomolies. Through the years researchers discovered, classified, and created solutions for these anomolies. These solutions, which involve improved criteria for designing relations, are called normal forms.

Codd has defined the first, second and third normal forms [Ref 14]. Later, Boyce-Codd normal form was postulated, and fourth and fifth normal forms were defined. Higher order normal forms are refinements of lower normal forms, so the higher forms are subsets of the lower forms. This is illustrated in figure 23 below.

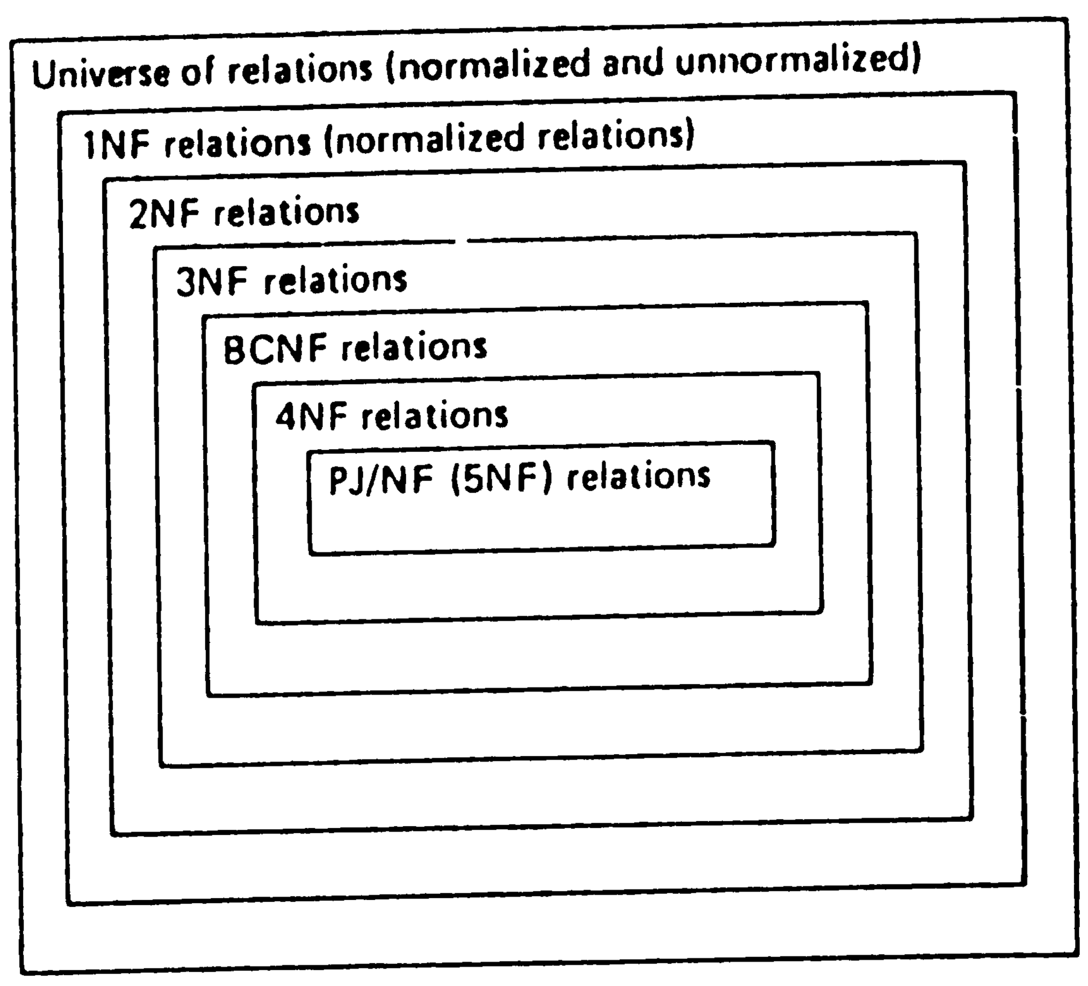


Figure 23 : Normal Forms

[Ref 1, pp 363]

Hence, a relation in fifth normal form is also in first, second, third, Boyce-Codd, and forth normal forms.

These normal forms are helpful but they a serious limitation. No one could guarantee that these would eliminate all anomolies. In 1981 R. Fagin defined a new normal form called DOMAIN/KEY normal form. He showed that a relation in this normal form was free of all modification anomolies. Unfortunately, it has not been proven that all designs can be placed in this form, so the existing normal forms are still used as design criteria.

First normal form is the easiest to satisfy because it is the broadest. A relation is in first normal form if all occurrences of that record type contain the same number of data fields and none of these fields contain the same type of information, called repeating groups. Repeating groups are not acceptable in relational theory. For example, figure 24 shows EMP-JOB1 relation where employees may have up to three jobs assigned, a repeating group.

EMP-JOB1 Relation

EMPNO	JOB	JOB	JOB	JOB
1111	965	983	999	
2222	984			
3333	912	913		

EMP-JOB2 Relation

EMPNO	JOB
1111	965
1111	983
1111	999
2222	984
3333	912
3333	913

Figure 24 : First Normal Form

To eliminate this first normal form violation EMP-JOB2 relation is created. Now, there are only two data fields and each record is the same length.

Second and third normal forms involve the relationship between the key and non-key data fields. The second normal form is violated when a non-key data field is a fact about only part of a composite key. Figure 25 shows the DIV-DEPT1 relation where the division address only provides a fact about the division number, not the division number and department number. Data duplication occurs here. The DIV-DEPT2 and DIV-ADDR relations show this second normal form problem corrected.

DIV-DEPT1 Relation

DIVNO	DEPTNO	DIVADDR
12345	100	200 West Packer
12345	200	200 West Packer

DIV-DEPT2 Relation

DIVNO	DEPTNO
12345	100
12345	200

DIV-ADDR Relation

DIVNO	DIVADDR
12345	200 West Packer

Figure 25 : Second Normal Form

The third normal form is violated when a non-key data field is a fact about another non-key data field. This is very similar to second normal form, and it is resolved in a similar manner. In general, when a relation is in third normal form, every data field is either part of the key or provides a single valued fact about the whole key.

The third normal form has limitations when a relation has more than one candidate key. Modification anomolies arise with this form when it is used with these relations. A subsequent normal form, know as Boyce-Codd, is more restrictive than third normal form and is used to resolve these problems. This form was not used for the database design, and it will not be discussed further.

The fourth and fifth normal forms came about to solve problems involving multivalued facts. These multivalued facts occur when data entities have many-to-many and many-to-one relationships. These too were not used for the database design, and they will not be discussed further.

Once a database has been normalized it may be free of modification anomolies and inconsistencies. This will minimize maintenance problems for a database that is frequently updated. However, this normalized design is not always beneficial. Retrieval can be expensive, since many relations may need to be accessed where only one needed to be accessed before. The correct extent of normalization should depend on the number of retrievals expected and the desired response time. Should a shorter response time be desired, some amount of data duplication can be incorporated.

IV.3 - Summary

The relational data model is a relatively new but important development in data processing. The model separates the users from the data implementation, freeing them from using very specific access techniques. Its table organization is unique and it provides many unique properties. The relational algebra operators are used to manipulate these tables and rearrange the data into forms desired by the users.

The design of the relational database can be a highly complex interrelated group of tables. Modification anomolies can occur during the data manipulations that can destroy the integrity of the database. Following normalization techniques and considering the usage characteristics of the database, a design can be achieved that provides a high level of effectiveness and efficiency.

Now that the relational model and its properties has been described in greater detail, the specific database manager used, ORACLE, will be highlighted. The database development task will then be discussed, and a critique of the system will be outlined.

V ORACLE

V.1 - Introduction

Having described relational database concepts, the specific relational database manager used for the database development is now discussed. The history, importance and an overview of the ORACLE system are given. The selection of this manager is justified and its main components outlined with particular attention given to its Structured Query Language.

V.2 - History and Overview

ORACLE is a relational database management system plus a complete set of software productivity tools that run on IBM mainframes, DEC and several other manufacturers' minicomputers and microcomputers [Ref 17, pp 1]. These software products were developed and are distributed by Oracle Corporation of Belmont California. The corporation was founded in 1977 with the goal of implementing the relational database "blueprint" produced in 1976 by the

IBM San Jose research facility. The goal was achieved in 1979 when ORACLE version 1 was released as the first commercial relational database in the world [Ref 16, pp 1]. No other relational database became commercially available until 1981 when IBM released its SQL/DS product.

Since its early release, ORACLE has enjoyed substantial market and technology leads in relational database products. To date, ORACLE is in use on over 1,000 IBM mainframes and DEC VAX minicomputers [Ref 17, pp 1]. In fact, ORACLE was the first DBMS to utilize Structured Query Language (SQL). This language has since become the industry standard and Oracle Corporation continues to push its boundaries (more will be discussed on SQL in a later subsection). Its version of SQL is a superset of IBM's DB2 and SQL/DS, and is completely compatible with these two. This is one reason why it remains the most powerful relational DBMS today.

There are numerous other features of the ORACLE RDBMS that provide attractive benefits. The software will run on equipment by IBM, Amdahl, DEC, Data General, Prime, Hewlett-Packard, AT&T, Sperry, Stratus, Honeywell, Harris, Wang, NCR, Pyramid, Sun, Apollo, Nixdorf, Olivetti, Toshiba, Hitachi, Fujitsu and Siemens. With this wide variety of hardware to choose from, ORACLE can be used as a corporate standard for all departments. The code is completely portable, and as computing demands change the ORACLE applications can be moved from one machine to another with no modifications.

ORACLE provides a distributed architecture that allows data and applications to reside on many computers and communicate without special end-user actions. This produces an environment where the computer resources can be utilized to their best advantage. This open system distributed DBMS architecture provides one standard logical database regardless of physical locations. Private databases can be maintained separately and access to public domain data can still be achieved by everyone.

Along with ORACLE's one logical view of the database, it provides efficient support of concurrent users. Applications can be used by many users at the same time. Frequently used programs are shared so memory and I/O costs are reduced. Users accessing the same relation are not locked out from data unless one user is updating the specific tuple of interest. This helps increase all users efficiency because responses are faster.

Two other features, an active data dictionary and a security facility, are major contributors to the RDBMS's success. The data dictionary is constantly active, dynamically maintaining an accurate reflection of the database contents as table definitions are modified. In addition to password protection, the security facility provides control of data access, manipulation, and creation right down to individual fields.

In addition, there are numerous other extended features, most of which cannot be found in other relational products:

- o Handling of null data values
- o Automatic recovery from crashes and media failure
- o Dynamic index maintenance
- o Query optimization
- o Distributed queries
- o Full support for views and joins
- o Outer joins
- o Date datatype support
- o Import/export of database data

[Ref 16, pp 2]

V.3 - Why ORACLE?

The ORACLE database management system meets all the necessary requirements for the tracking portion of the CIM system.

established

The DBMS is a completely established and proven product in use at many installations. This provides assurance that the product is stable. The code is free of major faults, and technical support for the product is in place and ready for trouble shooting and development suggestions.

multi-user support

Operators on the shop floor will be accessing the database with a high chance of concurrent use. Several foremen may also wish to check status at the same time. ORACLE allows all these users to simultaneously use the database while protecting against updating problems.

view support

Operators should not have access to all the same data that the foremen do. Even foremen should not be allowed to manipulate database information that is not pertinent to the suspension cell or information belonging to the system tables. ORACLE provides for the limiting of data by allowing views of the database to be defined for each user class. Only the DBA has an uncensored view of the entire database that permits him total control.

security

Password protection is essential to prevent unauthorized users from entering the system. In addition to this facet, ORACLE also provides security measures that grant different data capabilities for specified data items and users. This ensures that users on the system may be able to view some data but will not be able to alter the data without expressed permission.

import/export

The connection of the MicroVAX to the IBM 4381 will provide a link for tracking data to be transferred from the ORACLE database in the cell controller to the Cullinet scheduling system in the remote mainframe. This data must be extracted from the ORACLE database and presented in a form that the scheduling system will understand. The import/export facility ORACLE provides will dump database data to flat files readable by the IBM machine. Data created on the IBM in the scheduling system can also be put in flat files and imported to the database on the MicroVAX using this facility.

future expandability and user friendliness

All organizations are in a constant state of flux and must consider their future computing needs. This is not an easy task, and accurate predictions are rarely made suggesting that flexibility is a good directive. ORACLE provides a large hardware selection and allows databases to be distributed across these different machines. Its relational and state of the art status supplies easy and powerful support for applications. New directions in corporate policies or needs that result in new data processing requirements can be accommodated with this RDBMS.

V.4 - Components

The ORACLE system is composed of three major components: the Structured Query Language, the data elements dictionary, and the kernel (see figure 26). The

Structured Query Language contains several subparts, and it is discussed further in the next subsection.

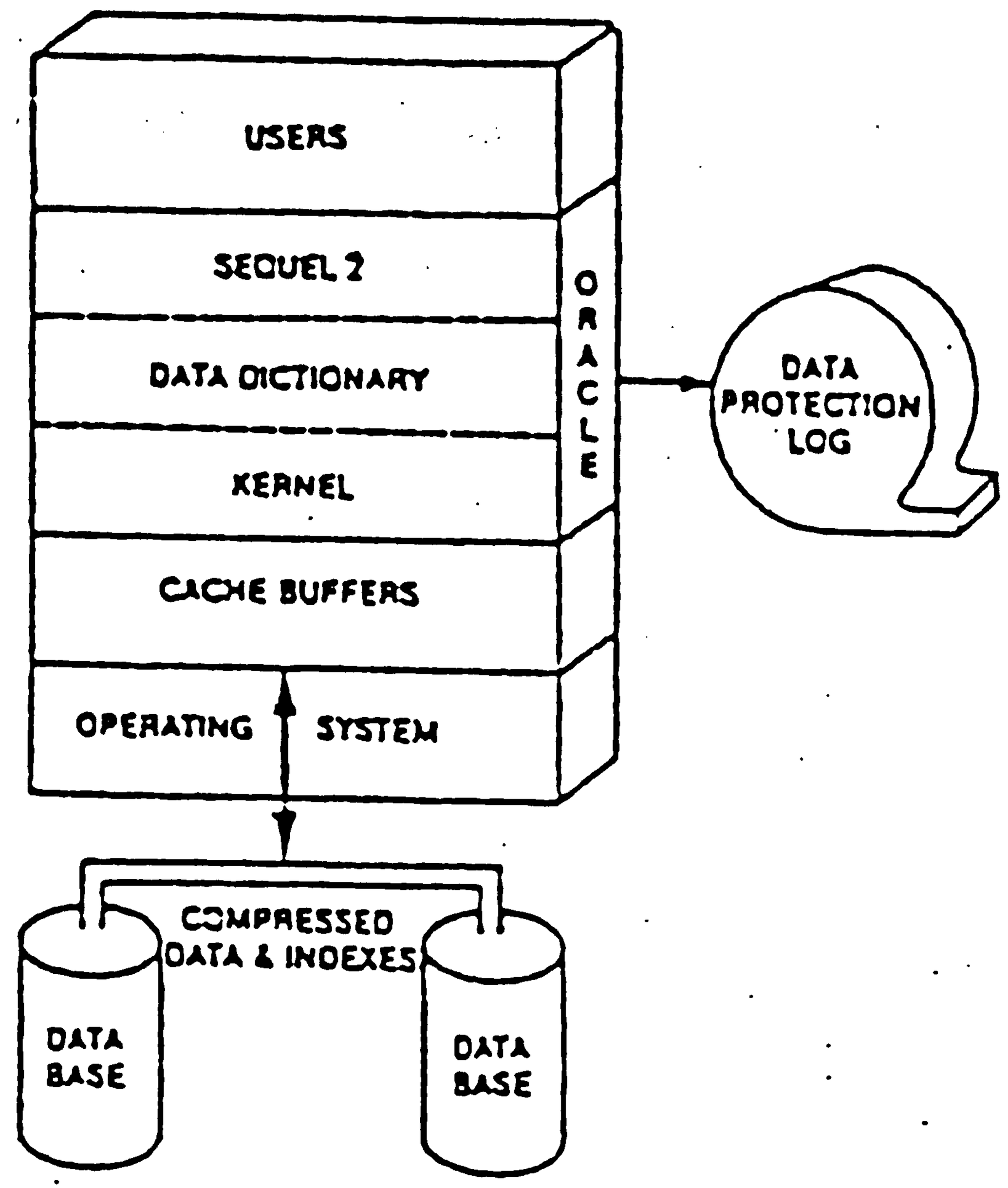


Figure 26 : ORACLE Components

[Ref 10, pp 112]

The fully integrated data elements dictionary is a very powerful tool for database designers. This dictionary contains table, row, and column definitions, views of tables and data fields, and information about users and their data access privileges.

The table, row, and column definitions are assigned dynamically when they are created. Any modifications to the database are automatically entered without the need for a reorganization of the database or a recompilation of existing programs. The dictionary also keeps track of the views of all tables, who created the view, and who has access to it. All security and control information on data and users is kept in the dictionary.

The kernel component is the core of the DBMS. It allocates resources for ORACLE in a similar manner that the operating system allocates resources for the computer system. The following paragraph provides a brief description of the kernel:

> "The kernel automatically reallocates and reuses space from a central storage pool to optimize the physical location of related data items within the data base. In addition, the kernel dynamically manages all ORACLE buffers, using an LRU (last record update) caching algorithm to maximize reuse of core-resident information. The kernel also enables ORACLE to support multiple concurrent batch and on-line terminal updates and queries to the data base. It can overlap I/O operations up to the limit of the hardware configurations. A single copy of the kernel can support simultaneous activity on separate independent data bases."
> [Ref 10, pp 112-113]

V.4.1 - Structured Query Language

Structured Query Language (SQL), sometimes referred to as SEQUEL 2, is an English-like language that supports five functions: query, data definition, data manipulation, data control, and host language coupling. The

English-like characteristics of SQL are used to create non-procedural queries of the database. The user describes what he wants using English terms, such as shown in figure 27, and ORACLE formulates a procedure to find the desired data.

```
SELECT JOBNO, PRIORITY
FROM JOBS
WHERE STATUS = 'C';
```

Figure 27 : A Non-procedural Query

query

The query function relies on an operation called mapping for its success. To illustrate, in figure 27 a known quantity (STATUS = C) is used to find a desired quantity (JOBNO and PRIORITY) from a relation (JOBS). Note that this query is actually a projection since it requests several attributes, or columns, of the JOBS relation. The three basic terms used for queries are SELECT, FROM, and WHERE. SELECT declares the desired information, FROM gives the relation(s) to be searched, and WHERE provides the known fact that must be satisfied.

data definition language

The data definition language (DDL) is used for defining tables, rows, columns, views, and indices. This DDL is a tool that is used by the database designers because it alters the actual database framework. All data structures must first be defined with DDL before they can be accessed by other SQL facilities. Figure 28 gives an example of a DDL statement.

```
CREATE TABLE JOBS
(JOBNO NUMBER(5) NOT NULL,
PRIORITY CHAR(1),
STATUS CHAR(1));
```

Figure 28 : Data Definition Example

The number in parentheses is the maximum number of spaces that the data field can occupy, and NOT NULL specifies that the field is mandatory.

data manipulation language

The data manipulation language (DML) allows users to change the value of data stored in the database. Three operations can be performed: insert, delete, and update. Figure 29 provides an illustration.

```
INSERT INTO JOBS
VALUES (12345,'A','O');

DELETE FROM JOBS
WHERE JOBNO = 77745;

UPDATE JOBS
SET STATUS = 'C'
WHERE JOBNO = 12345;
```

Figure 29 : Data Manipulation Examples

An insertion is used to insert new data into a table or to extract data from one table and place it in another. A deletion is used to delete a row or rows from a table, and an update is used to change the value of one or more data fields in either a single or multiple rows.

data control

The data control language (DCL) enables the user to specify who will have access to his data and to enforce data integrity constraints. The user who creates a table owns it and has the authority to designate who else can interface with that table and the data stored there. The following are the privileges that can be granted by the owner of a table [Ref 7, pp 20.10]:

SELECT	data in a table or view
INSERT	rows in a table or view
UPDATE	values in a table or view

DELETE	rows from a table or view
ALTER	column definitions in a table
INDEX	index to a table

The table owner can pass any of these privileges to selected users or all users if the PUBLIC option is used. If the owner specifies the GRANT OPTION when he grants privileges, the recipient of the privilege can grant that privilege to others. Figure 30 gives an example of a DCL statement.

```
GRANT SELECT
ON JOBS
TO USER10
WITH GRANT OPTION;
```

Figure 30 : Data Control Example

host language interface

The host language interface allows SQL statements to be included in high-level language application programs such as COBOL, FORTRAN, C, Ada, PL/I, and Pascal. Instead of coding all the necessary complex statements in an application program to access the desired data, the high level SQL statements are coded. The applications are then run through a precompiler that will translate the SQL statements into the appropriate statements for the programming language used. This proves to be a major productivity benefit in system development phases.

V.5 - Summary

The ORACLE RDBMS has always maintained a lead in relational technology and remains the most powerful relational database system commercially available today. This system provides many substantial features such as a distributed architecture that allows utilization of many vendors hardware and still maintains one logical view of the database. Multi-user support, an active data dictionary and comprehensive security facilities are also significant facets.

The products that comprise the system provide these benefits and many more such that every need for the CIM system is reckoned. The product is established so that unexpected software glitches should be minimal. It will support multiple concurrent users allowing many operators and foremen to access data at the same time. In addition, the database can be partitioned into views that separate classes of users such as foremen and operators. The security features ensure database integrity by precluding unauthorized data manipulation. Data can be exported from, and imported to the database for used in other systems, and the expandability supplies flexibility for the future.

ORACLE's best feature is its use of Structured Query Language. This high-level language provides a user friendly interface to the system that supports various functions. The language allows for querying the database, defining data structures, manipulating data in the database, controlling data, and imbedding commands in high-level application programs. The support of these functions is very comprehensive and is the basis of the RDBMS's power and dominance.

Now that the specific relational database manager has been described, the approach used for developing the database will be outlined followed by a critique of the tracking system using the database.

VI DATABASE DEVELOPMENT

VI.1 - Introduction

Now that there is an appreciation for the relational database model and the specific RDBMS used, the development of the database must be explained. This includes the steps taken to create the overall conceptual schema and its transcription into the actual database model.

VI.2 - Database Design

The design of the database can easily be argued to be the most important step in the development of a computerized information system. The design must produce a database structure that is both "good" and "right." Good connotes a design that is efficient or at least meets some performance standards, and right connotes a design that meets the requirements of the functions it supports. Usually this process is very time consuming because of its importance and its many components. Development of a poor design creates a poor system that is only partially utilized, or worse, a system that must be redesigned at considerable expense.

Developing the database is a process that is evolutionary, with the final objective being an "idealized database." This database should contain all the data necessary to create the required information about all facets of an organization's operations. The information should be extractable quickly and in any form desired.

VI.3 - Schema Design

A relational database is specified by a relational schema which is the collection of the relational subschemas. Each relational subschema is a listing of a relation name and its corresponding attributes. Figure 31 gives an example of a relational subschema for the tracking system.

READYJOBS (JOBNO, WCID, PARTNO, NHANO, QTYRQD, EMPNO, RECDATE, PRIORITY)

Figure 31 : Relational Subschema

There are four steps involved in the design of relational subschemas.

1) Determine the information requirements for the different areas of the organization involved in the schema design.

2) Express the information requirements as an enterprise description.

3) Obtain a database description that more rigorously defines the database structures and constraints, and satisfies the information requirements.

4) Check the schema for the performance requirements of the prospective users.

VI.3.1 - Requirements Analysis

Requirements analysis is a high-level analysis of the functions of an organization or subgroup. The purposes of this analysis are to:

- Understand the operations of the proposed area.
- Determine the information requirements of the area by observing the means of doing business.
- Represent the requirements with a formal modelling technique.

How the area operates and what information is required to operate are the main questions to be answered here. A document should be produced that records the information flows that exist.

Information is the underlying force that drives all businesses. Without appropriate information and information flow, business functions cannot be adequately performed. Tracing and documenting the information that flows within an organization or subgroup is an excellent means for understanding the business.

The information flows in the suspension cell were followed and documented using a technique called structured analysis. Structured analysis is a top down approach to modelling the data flows in some bounded functional area. It dictates an iterative and systematic method of analyzing the data by defining the tasks and processes performed and breaking them down until they can no longer be reduced. Once the structured analysis is completed, function and data classes can be identified.

A function in an organization is an essential activity or decision required to manage the resources and operations of the organization. The following functions were identified for the suspension cell: employee time reporting, job time reporting, material tracking, quality assurance, and job status checks.

A data class is an aggregation of data required by a function or produced by it. The functions and some related data classes are shown in figure 32.

#	Function
1	employee time reporting
2	job time reporting
3	material tracking
4	quality assurance
5	job status checks

DATA CLASS	FUNCTION				
	1	2	3	4	5
EMPLOYEE NUMBER	*	*			*
EMPLOYEE NAME	*	*			
IN TIME	*				
OUT TIME	*				
JOB NUMBER		*	*	*	*
WORKCENTER ID		*	*	*	*
RECEIVED DATE		*	*		*
PRIORITY			*		*
START SETUP	*	*			*
FINISH SETUP	*	*			*

Figure 32 : Data Classes and Functions

were identified for the suspension cell: employee time reporting, job time reporting, material tracking, quality assurance, and job status checks.

A data class is an aggregation of data required by a function or produced by it. The functions and some related data classes are shown in figure 32.

#	Function
1	employee time reporting
2	job time reporting
3	material tracking
4	quality assurance
5	job status checks

DATA CLASS	FUNCTION				
	1	2	3	4	5
EMPLOYEE NUMBER	*	*			*
EMPLOYEE NAME	*	*			
IN TIME	*				
OUT TIME	*				
JOB NUMBER		*	*	*	*
WORKCENTER ID		*	*	*	*
RECEIVED DATE		*	*		*
PRIORITY			*		*
START SETUP	*	*			*
FINISH SETUP	*	*			*

Figure 32 : Data Classes and Functions

VI.3.2 - Enterprise Description

Creating the enterprise description consists of four phases. The first phase identifies the entities of interest to each organizational area, their view. The entities are described and their associated data classes are grouped with them. The second phase identifies the relationships that exist between entities. The third phase identifies the constraints on the attributes, entities, and relationships. Phase four then integrates the individual area views into one enterprise description, the entire conceptual schema.

VI.3.3 - Database Description

This step transforms the enterprise description into a database description. Now the enterprise description is used to describe the model to the target DBMS. Entities are mapped into relations permanently stored in the database. The result is a documentation of the schema into a database model. The last step is to test the design to determine if is good and right.

VI.4 - Summary

Relational database development is an extremely important task aimed at producing an "idealized database" that is both good and right. The development of the schema must go through several time consuming steps in order to finally reach a state where they are ready to be incorporated into a database model. Once the model has been established, it should be tested to assure the design meets all the users needs and the performance is acceptable.

With database development now described, the tracking system will be critiqued. Findings and future implications will then be given.

VII SYSTEM CRITIQUE

VII.1 - Introduction

The database development created a database model for use in the tracking portion of the CIM system. This tracking system will now be reviewed. The support the system provides is shown and its accomplishments, weaknesses and limitations are outlined.

VII.2 - Tracking System Support

The database model supports the five functions identified in the requirements analysis. Each function is supported by the various means described below.

employee time reporting

The system holds the employee clock-in and clock-out data. Reports can be generated that show daily employee time on the job and a breakdown of where each portion of that time was spent.

job time reporting

Operators at each of the machine tools input the time for the start of setup, finish setup, start job, finish job, and any extraneous time allotments that will be charged to a job. Reports can be produced that show actual time spent on each segment of a job.

material tracking

When material is delivered to the far end of the suspension cell it is recorded in the database. As the material is moved from various workcenters to inspection and finally to the opposite end of the cell, its location is updated in the database.

quality assurance

Inspection stations require the archival of rejected and accepted numbers. This data is also stored in the database.

job status checks

Each job maintains its own tuple in the database. This stores the data that shows when the material is ready for a job, when the job was started, what stage the job is in, and the number of scrapped and nonscrapped parts.

VII.3 - Accomplishments

The tracking system provides many benefits and accomplishments. Each of the five functional areas exhibit tangible accomplishments along with the general accomplishments for the whole cell. These will be discussed below.

employee time tracking

With the implementation of the database, employees can now clock in and out at any of the IDS stations located on the shop floor instead of only at the two time clocks. Time cards from the previous manual system are eliminated, and all the tedious and time consuming collation of this time data is taken care of by the database. Foremen can now access the database and find out when each employee arrived and departed for each day. The break down of employee's time on the job is also accessible. Queries can be written to produce exception reports that, for example, retrieve the employee names which have the most amount of unproductive time.

job time reporting

Time spent on each job can now be accounted for more accurately. Employees simply input the start and stop times when new phases of a job begin. There can be no unaccounted for time, so actual time spent on a job will be recorded. This means that job costing can be done directly from the stored database data, and more accurate costing will result.

material tracking

Previous to the database tracking system, material was not tracked in the suspension cell other that by employees searching for material as it was needed. Instead of foremen traveling to the material receipt end of the cell to determine what material has arrived, material is logged into the system when it is delivered to the cell. Now foremen can check the database to determine what material is available to start which jobs. This is a significant aid in initially assigning jobs.

Material can also be tracked as it moves through the cell. Once operators complete a job the material is moved to the next machine site and the location of the material is then updated in the database. Accessing the database will show all the current locations of material, hence, material is not lost and subsequent jobs can begin in a more timely manner.

quality assurance

Quality assurance inspectors can input the number of accepted and rejected parts to the database as soon as they have determined the appropriate numbers. With this updated job information, the effect on final orders can be seen quicker. More parts can be scheduled for production more rapidly to avoid delays in order completion.

job status checks

Management can monitor the status of jobs quickly and easily from any IDS attached to the system. Material location, time spent, and number of rejected parts can all be determined on-line whenever desired.

general benefits

With this system, the manual data flows that were the only means of passing information have been drastically reduced in number. This means paper forms which can be lost or destroyed are eliminated as the suspension cell moves towards the status of the paperless factory of the future.

Increased security is also a benefit. Passwords prevent unauthorized users from accessing the database, and authorized users can only access their specified view of the data.

Should the shop floor layout change, the system can accommodate the changes. The system is flexible. By simply moving IDS terminals to the new workcenter locations, the system is ready to function in the new environment. The system is also expandable. More IDS stations can be added for each new workcenter added, up to the limit of terminals the cell controller can handle.

Inherent in all database systems is the possibility for querying the database to answer ad hoc questions such as the employees with the most amount of unproductive time mentioned earlier.

VII.4 - Weaknesses and Limitations

While the tracking system does provide many benefits, there are weaknesses and limitations that must be considered. Database weaknesses, system specific limitations, and general limitations are the areas that will be discussed below.

database weaknesses

The three disadvantages of databases given in chapter III all apply as weaknesses to the tracking system.

The central nature of the system places the suspension cell in a vulnerable position. Should the DBMS software or the cell controller hardware fail, the cell is left without a tracking system until the malfunctions can be corrected. In systems, it is beneficial to build in some redundancy in case of a failure. A manual, paper system should be in place ready for use just in the event of a failure.

Cost and complexity are also factors to consider when using databases. The system must be maintained. When relations increase in the number of tuples they hold, they can become so large that they occupy too much memory and seriously degrade system performance. Personnel, such as a DBA, must be trained to maintain the complex software components. This coupled with the initial price of the DBMS can produce high costs. These should be compared to the possible productivity improvements in a cost/benefit analysis.

system specific limitations

The design of the CIM system and its subset tracking system include areas that could be improved.

Each of the IDS machines on the shop floor only have one monitor. This produces a limitation in that only one screen may be displayed at any certain time. An operator who is examining a part drawing on the monitor must close the part drawing file if he then wants to access the database. A multitasking, dual monitor display station would be a better solution.

Data entered into the database can be checked to determine if it is a valid input for that field (does the data fall within a specified domain?). But erroneously entering the wrong value that is still in the domain will go undetected. For example, if an operator enters a valid yet incorrect workcenter number as the destination to where he delivered material, the system will have one location listed for the material, but the material will actually be at another location.

Foremen can check the status of jobs but only with a limited capacity. They can tell what major phase a job is in such as setup or running, but they cannot determine what percentage of parts on a job have been completed. The DNC system keeps track of this data, and it is not integrated with the database.

general limitations

Two general limitations have been identified.

The whole CIM system was only implemented on one machine cell within a large manufacturing firm. When the system is expanded to include other cells, the performance characteristics will change. Most likely, the responses when accessing the database will degrade from overuse of the resource. In addition, as the system is integrated with others, some type of governing control system will be required.

The last limitation identified also pertains to integration. The data stored in the database such as job status should be used in other systems for functions such as Manufacturing Resource Planning. The planned future expansion to the IBM 4381 and the Cullinet software is a good example of what should be done.

VII.5 - Summary

The tracking system supports all the functions identified in the requirements analysis with numerous benefits that increase the productivity of the suspension cell. However, as with any system, there are areas where the system can be improved. In general the system provides substantial benefits that outweigh the weaknesses.

The final chapter supplies the findings and areas for future study.

VIII CONCLUSIONS

VIII.1 - Introduction

This research was conducted for the sole purpose of learning about relational databases and CIM. The major findings during the research and implications for the future of relational databases in CIM are presented here.

VIII.2 - Findings

Database management systems are an extremely powerful software tools that permit centralized control of an organization's operational data along with a host of spawned benefits from this control. These systems have been in use in data processing for years and are now gaining more and more use in the realm of manufacturing.

One database model, the relational model, is becoming extremely popular because of its unique separation of the users from the low-level data organization. It facilitates the creation of queries in a high-level, nonprocedural, English-like language.

The ORACLE relational database management system used for the development of the tracking system is the most advanced RDBMS available commercially. ORACLE employs a nonprocedural language called Structured Query Language (SQL). This language is very powerful and supports the very important functions of querying, data definition, data manipulation, data control, and host language coupling through the use of its user friendly commands.

The relational database development is the most important task to be performed when designing a system. It is critical that the final design is both good by being efficient, and right by meeting the functional requirements. There are methods for development that take into consideration the properties of relational databases. The method used here satisfied both the good and right criteria.

The tracking system created provided many benefits to the suspension cell's operation. Each of the identified functional areas was supported and improved by the system.

There were also weaknesses and limitations to the system. Weaknesses with databases in general such as vulnerability, cost, and complexity can be minimized but not eliminated. Application specific limitations can be addressed differently, and many will be discussed in the next section.

VIII.3 - Future Study

CIM encompasses many factors in manufacturing, but its key aspect is the integration of components into a system. Integration implies that there is effective communication between the constituents to provide each with the data it needs, when it needs it. With the implementation of the tracking system, weaknesses in communications became apparent.

The planned future expansion connecting the IBM 4381 to the MicroVAX cell controller will be a welcome system upgrade. This will permit the Cullinet scheduling system on the IBM to access the status data in the MicroVAX's database. Another function should be included. The scheduling data should be downloaded to the database so the schedule can be accessed on-line instead of on paper.

Another area for future development is the DNC system. This system is completely separate, maintaining its own files, from the database tracking system. This imposes a division on sharing the data that should be removed. With both the tracking and DNC systems accessing the same database, more useful information can be produced from the collective pool of data. For example, how many parts have been completed from the current batch, and how much time it takes to run one part.

The database is an expensive resource and should be utilized to its fullest capacity. Supplemental applications could be written that produce management reports comparing planned to actual progress on jobs in the shop. Analysis should be performed on stored data to evaluate cell performances in different shop floor layouts. The results from this analysis could be used to design future cell layouts.

One last area identified for future work is the application of larger database systems installed to support larger manufacturing facilities. As systems grow, there will be a trade off between the benefits of sharing data in a centrally controlled facility, and the decrease in effective responses to data requests from an over burdened database. Distributing the database and still maintaining control in a manufacturing setting would provide a great step forward in establishing the truly integrated factory.

VIII.4 - Implications

Relational database and CIM technologies are both progressing at a rapid pace. It becomes clearer with each CIM endeavor that to manage the vast amounts of data in manufacturing, data processing tools such as relational database systems are required. Both individually hold great promise for increasing productivity. The marriage of these two provide even greater potential for manufacturing competitiveness.

LIST OF REFERENCES

[1] Date, C. J.. An Introduction to Database Systems, Fourth Edition, 1986, Addison-Wesley Publishing Co.

[2] MacDow, Rodney and Dirk, Alan. Managing Multiple Databases in CIM, 1985, A paper from Prime Computer, Inc. Natick, Massachusetts.

[3] Manufacturing Studies Board. Commission on Engineering and Technical Systems, National Research Council, Toward a New Era in U.S. Manufacturing *The Need for a National Vision*, 1986, National Academy Press, Washington, D.C..

[4] Kroenke, David. Database Processing, Second Edition, 1983, Science Research Associates Inc., Chicago Illinois.

[5] Kaplan, R. S.. "Must CIM be Justified by Faith Alone?", Harvard Business Review, March-April 1986.

[6] Oracle Corporation. ORACLE for DEC VAX/VMS Installation and User's Guide, Version 5.1, 1986, Oracle Corporation, Belmont, California.

[7] Sachs, Jonathan. SQL*Plus User's Guide, Version 2.0 1986, Oracle Corporation, Belmont, California.

[8] Dimmick, Shelly. ORACLE Database Administrator's Guide, Version 5.1, 1986, Oracle Corporation, Belmont, California.

[9] Zussman, J. U., Sachs, Jonathan and Gomez, Gregory. SQL*Forms Designer's Reference, Version 2.0, 1986, Oracle Corporation, Belmont, California.

[10] Weiss, Harvey, M.. "The ORACLE Data Base Management System", MINI-MICRO SYSTEMS, August 1980.

[11] "A Drastic New Loss of Competitive Strength", Business Week, June 30, 1980.

[12] Lodge, G. C. and Crum, W. C.. "U.S. Competitiveness: The Policy Tangle", Harvard Business Review, January-February 1984.

[13] Benjamin, R. L., Rockart, J. F., Morton, M. S. and Wyman, J.. "Information Technology: A Strategic Opportunity", Sloan Management Review, Spring 1984.

[14] Codd, E. F. "A Relational Model of Data for Large Shared Databases", In Communication of the ACM, Vol. B, No. 6, June 1970.

[15] Conte, Paul. "Understanding Relational Data Bases", Computer Language, Vol. 4, No. 5, May 1987

[16] Oracle Corporation. ORACLE Products and Services Overview, 1986, Oracle Corporation, Belmont, California.

[17] Oracle Corporation. ORACLE Overview and Introduction to SQL, May 1985, Oracle Corporation, Belmont, California.

VITA

Ronald Habakus was born in Johnstown, Pennsylvania to William Stephen and Louise Marie Habakus on October 18, 1961. In 1963, the family relocated to Bethlehem, Pennsylvania.

Ron graduated from Freedom Senior High School in Bethlehem in 1979. He attended Lehigh University that fall and enrolled in the College of Engineering and Physical Sciences. In June of 1984, he graduated with a Bachelor of Science degree in Industrial Engineering and minors in Psychology and Computing Science.

Upon graduating, Ron worked as a Management Information Consultant for Arthur Andersen & Co. in Washington, D.C.. In November of 1985, Ron was offered an assistantship at Lehigh University, and in January of 1986 he accepted the offer and began a Master of Science degree in Manufacturing Systems Engineering. He is currently finishing this program and will graduate in June of 1987.

RETAKE

The Operator has Determined that the Previous Frame is Unacceptable and Has Refilmed the Page in the Next Frame.

[11] "A Drastic New Loss of Competitive Strength", Business Week, June 30, 1980.

[12] Lodge, G. C. and Crum, W. C.. "U.S. Competitiveness: The Policy Tangle", Harvard Business Review, January-February 1984.

[13] Benjamin, R. L., Rockart, J. F., Morton, M. S. and Wyman, J.. "Information Technology: A Strategic Opportunity", Sloan Management Review, Spring 1984.

[14] Codd, E. F. "A Relational Model of Data for Large Shared Databases", In Communication of the ACM, Vol. B, No. 6, June 1970.

[15] Conte, Paul. "Understanding Relational Data Bases", Computer Language, Vol. 4, No. 5, May 1987

[16] Oracle Corporation. ORACLE Products and Services Overview, 1986, Oracle Corporation, Belmont, California.

[17] Oracle Corporation. ORACLE Overview and Introduction to SQL, May 1985, Oracle Corporation, Belmont, California.

VITA

Ronald Habakus was born in Johnstown, Pennsylvania to William Stephen and Louise Marie Habakus on October 18, 1961. In 1963, the family relocated to Bethlehem, Pennsylvania.

Ron graduated from Freedom Senior High School in Bethlehem in 1979. He attended Lehigh University that fall and enrolled in the College of Engineering and Physical Sciences. In June of 1984, he graduated with a Bachelor of Science degree in Industrial Engineering and minors in Psychology and Computing Science.

Upon graduating, Ron worked as a Management Information Consultant for Arthur Andersen & Co. in Washington, D.C.. In November of 1985, Ron was offered an assistantship at Lehigh University, and in January of 1986 he accepted the offer and began a Master of Science degree in Manufacturing Systems Engineering. He is currently finishing this program and will graduate in June of 1987.