Design & Development of a Database as a Part Data Management System

A Thesis submitted to the Division of Graduate Studies and Research Of the University of Cincinnati In partial fulfillment of the Requirements for the degree of

MASTER OF SCIENCE

In the Department of Mechanical, Industrial and Nuclear Engineering Of the College of Engineering 2000

By

Chirag Belosay

B.S., (Production Engineering), Bombay University, India, 1997

Thesis Advisor and Committee Chair: Dr. Ernest L. Hall
Abstract

A brief overview of data models and schemas followed by an introduction to the various types of databases, the main steps in database design and the various database languages available is presented. This thesis explains the data definition language and data manipulation language syntax and commands. It illustrates how structured query language evolved into the globally accepted relational database management system language [27]. It highlights the concepts behind the Entity Relationship diagram [11]. It bears a section on test analysis, improvements made to the database management system and the future of database management systems. The purpose of this was to implement a database design customized for robot part information, using a database management system. A test conducted to document the efficiency and usability of the database was highly positive. The results indicated the efficiency by which the data could be managed. The significance of this thesis is that it exposes the user to the advantages and benefits underlying the usage of databases and as to how one can leverage their business by implementing a DBMS.
Acknowledgements

It would be totally unprofessional on my part to claim complete credit for the successful culmination of this thesis. I therefore take this opportunity to express my deep gratitude and acknowledgements to all those individuals involved both directly and indirectly for their invaluable help and guidance.

I am grateful to Dr Ernie Hall for his constant support and encouragement. It is my pleasure to present this thesis a written testimony of the effort and hard work put in to complete this successfully. Last but not least I dedicate this thesis to my Parents who have always been there for me. Their support was a vital factor in my completion of this thesis.
# Table of Contents

**Abstract**

**Acknowledgement**

**List of Figures/Tables** ................................................................. 6

1. **Introduction & Overview** ............................................................
   1.1 Introduction ................................................................. 7
   1.2 DBMS what exactly is it? .................................................. 8
   1.3 Data Models. ................................................................. 11
   1.4 Schemas and Instances. .................................................. 12
   1.5 DBMS Interfaces .......................................................... 15

2. **Classification of Database Management Systems.** ..................... 17
   2.1 Flat file Databases. ....................................................... 17
   2.2 Network and Hierarchical Databases. ................................ 17
   2.3 Relational Databases. .................................................... 19

3. **Phases of Database Design** .................................................... 23
   3.1 Phases of Database design .............................................. 23
      3.1. a Step One (Model the Data) ..................................... 24
      3.1. b Step Two (Database Diagrams) ................................. 26
      3.1. c Step Three (Normalize the Data) ............................. 27
      3.1. d Step Four (Primary Keys) ..................................... 28
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. The Language and the DBMS.</td>
<td>33</td>
</tr>
<tr>
<td>4.1 Why SQL</td>
<td>33</td>
</tr>
<tr>
<td>4.2 DDL - Data Definition Language</td>
<td>34</td>
</tr>
<tr>
<td>4.3 DML – Data Manipulation Language</td>
<td>35</td>
</tr>
<tr>
<td>4.4 Why ORACLE?</td>
<td>36</td>
</tr>
<tr>
<td>5. The Building of the Database.</td>
<td>40</td>
</tr>
<tr>
<td>5.1 The ER Diagram.</td>
<td>40</td>
</tr>
<tr>
<td>5.2 Test Analysis and Improvements</td>
<td>44</td>
</tr>
<tr>
<td>5.3 Where do we go from here? – Future Scope.</td>
<td>48</td>
</tr>
<tr>
<td>6. Conclusion</td>
<td>51</td>
</tr>
</tbody>
</table>

References

Appendix
List of Figures

Figure 1. Database Systems.

Figure 2. Database Table Structure.

Figure 3. DBMS –3 Schema Architecture.

Figure 4. Relationship’s between Tables.

Figure 5. Phases of Database design.

Figure 6. Entity Relationship Diagramming Method.

Figure 7. Oracle Navigator Interface.

Figure 8. ER Diagram.

Figure 9. ER Diagram (Logical)

Figure 10. SQL Image – Tables Created.

Figure 11. Data types used.

Figure 12. GUI Interface.

Figure 13. GUI Interface.

Figure 14. Modified ER Diagram.

Figure 15. Closed loop Decision Support System.

Table A. Primary and Foreign Keys in a Table.

Table B. Relationships and connections across tables.

Table C. Test Analysis for Group A.

Table D. Test Analysis for Group B.
Chapter One

Introduction and Overview.

1.1 Introduction:

The Cincinnati Center for Robotics Research at the University of Cincinnati has been a participant in the AUVS (Association for Unmanned Vehicle Systems) competition for the last 7 years. The University of Cincinnati has so far used two versions of their AGV’s(Automated Guided Vehicles). The first, Bearcat I, robot weighs approximately 600 lbs., and is 4 feet wide and 6 feet long. In the 1997 competition, the track was 10 feet wide. This meant that the space left for maneuvering was 3 feet on either side of the robot. The increasing difficult competition rules made it necessary to build a new AGV. This resulted in the creation of the Bearcat II robot. In 1998 the UC Robot Team developed a new design for a mobile robot.

The most outstanding feature of the design was that the entire structure was designed, built and assembled by the student team within the UC lab. Most of the components were ordered from different vendors with the bulk of the body structure coming from the 80/20 extruded aluminum components, which were made to order. One of the problems anticipated and encountered by the robot team was part burnout or wear-out. The problem encountered by the robot team was in case of a part burnout or wear-out, that particular part would have to be reordered. This required locating the particular part, its dimensions, available vendors and then placing the order. The entire process was inefficient and would usually result in long delay periods.
This brought about the idea of developing a DBMS similar to a PDMS (Part Data Management System) to help manage the information about the various parts. A DBMS (Data Base Management Systems)\(^2\) is a collection of programs that enable us to create and maintain a database. Once this idea was conceived it was necessary to decide how should it be implemented. The first step was to get the design onto paper. This would be followed by the software modeling the concept of the database, which would eventually be followed by the running and testing of the model. Before we dive into the entire process of structuring and building of the DBMS, we need to understand certain basic concepts of database design.

This thesis is broadly divided into six main sections namely

1) What was the problem
2) Available Options to rectify the problem
3) Options Analyzed
4) Option Implemented
5) Testing and Analysis of the Implemented Solution
6) Future scope of the project.

1.2 DBMS - What exactly is it?

As indicated earlier, DBMS (Data base management systems)\(^2\) is a collection of programs that enable us to create and maintain a database. It helps in defining, constructing and manipulating a database. A database system is shown in figure 1
Defining a database involves specifying the data types and the various applicable constraints. Constructing involves entering the data into the memory, and manipulation includes various queries made to the database.

The intended use and capabilities of a DBMS are:
1) Controlling redundancy— Redundancy in storing the same data multiple times can lead to several problems such as duplication of effort, wastage of storage space, inconsistent data, difficulty in management and manipulation of the database. The DBMS software makes sure that the data entered is not redundant; the various triggers such as when validate item are activated when any redundant data is being inputted.

2) Restricting unauthorized access— In any database there are situations where a certain group of users are authorized for a particular portion of the data from the database, for example financial salary needs to be accessed by authorized persons only. The DBA (Database Administrator) can program the DBMS to restrict unauthorized access. A general security and authorization system can be easily programmed to prevent unauthorized access.

3) Providing multiple user interfaces - Various users are involved in accessing a database and running different queries. The DBMS helps provide different user interfaces such as query language for the casual user, programming language interface for the application programmer, forms for the parametric user and menu driven interface for the standalone user.

4) Providing backup and recovery— Backup and recovery is an essential requirement of a DBMS. In case of a system failure during the middle of a complex program it is essential that the database be restored to the state it was in, and the recovery system must ensure that the program can be effectively resumed.

5) Enforcing integrity constraints— Most databases require certain integrity constraints to be enforced. For example, a zip code must numerically be 5 digits long. The
integrity constraints are essential to ensure the accuracy of the data retrieval from the database.

6) Ability to represent complex relationships between data items - Any database can include a large amount of data, which can be interrelated in many possible ways. The RDBMS should be able to display these relationships effectively so it is able to assist the end user in understanding the various relationships existing in the data.

7) Storing Program objects and data structures - In most of the object-oriented languages, there are various data structures involved such as classes in C++. There are cases where the values of program variables are discarded once the program terminates, such as in PASCAL or C++. When the need arises to rerun the data again, the programmer must convert from the file format to the program variable structure. This conversion is done automatically by the DBMS software.

1.3 Data Models - A data model is basically a set of concepts used to define the structure of the database. The data types, relationships and the various constraints existing within the database define the structure of a database. Data Models can be classified into high-level data models and low-level data models based on the type of concepts they use to describe the database structure. Low-level data models are usually used for users who possess expertise in the database field, whereas the usual end users use the high-level data models. A class of representational or implementation models between the two extremities also exists. High-level data models usually bear concepts such as attributes, entities and relationships. Entities represent real world concepts; for example, an entity could be a student or an employee, whereas attributes represent certain
properties of entities and act as an adjective to the entity. Relationships represent interaction between the entities.

Consider the following example. Student A is identified by his social security number and he needs to take certain courses, but at the same time make sure that he has the necessary prerequisites. Therefore, in the above case, student A is an entity. His social security number is an attribute for the entity student, and the prerequisite condition is the relationship that exists. The generalized structure of a database table is given below

![Database Table Structure](image)

**Figure 2. Database Table Structure.**

1.4 Schemas and Instances - A database schema is the description of the database; it is also know as the Meta-data. The DBA sets up the schema for the database and usually the schema is the most constant portion of the database. The objects in the schema are called the schema construct. It should be noted that even though it is easy to display the database schema model easily using a schema diagram, it is however not possible to indicate all the existing constraints. The Database State is the factor that keeps changing every time any changes are made to the data. The Database State is the data in
the database at that particular instance. Each time new data is entered or some data is either updated or deleted, the Database State is changed from one state to another. Let us now discuss the 3 Schema Architecture.

Figure 3. DBMS Architecture, The 3 Schema Architecture
The 3 Schema architecture divides the DBMS into the following three levels

a) Internal level - This level represents the physical data, storage structure. The 3 schema uses a physical data model to illustrate the details of the data storage and access path.

b) Conceptual Level- This level hides the details of the physical storage and the access paths and highlights more on the entities, relationships, constraints and user operations. The conceptual level describes the structure of the whole database for a community of users.

c) External Level- This level consists of a number of user friendly views. Every end user need not look at the same type of data retrieval from the data base and that’s exactly what the external level does, it describes part of the database that the particular user is interested in and hides the remaining data.

The 3 schema’s are only descriptions of data, while the actual data exists at the physical level. The end user only refers to his own external schema and thus it is the function of the DBMS to covert the request by the end user. The data is then extracted from there and returned back to the end user. The process of transforming the requests between the different stages is called mapping.

Data Independence - Data Independence is the capacity to change the schema at one level without having to change the schema at the next higher level. The two main types of data independence are:

1. Logical data independence- The ability to change the conceptual schema without having to change the external view or the eternal schema’s. The conceptual schema may expand when there is an addition of an extra data item, or it may
reduce when a record type is deleted. When any of the above is done, the external
schema referring to the remaining data should not be affected.

2. Physical Data Independence- The capacity to change the internal schema without
having to change the conceptual or external schema. The internal schema may
change whenever there is a need to reorganize some of the physical files to
improve the retrieval capacity or efficiency of the database. Hence, when the
same data with better access structure exists, it should not affect the external or
conceptual schema.

Data independence is accomplished when a schema at one level is changed. It does not
affect the schema at the higher level, and only the mappings between the 2 levels are
changed. Thus, the 3-schema architecture really helps achieve data independence. The
one drawback of 3-schema architecture is that the two levels of mappings create an
overhead during the compilation and execution, thus rendering the DBMS inefficient.
Hence, although the 3-schema architecture allows you true data independence, very few
DBMS actually use the 3-schema architecture.

1.5 DBMS Interfaces- The front end for accessing the database are of the following
types.

- Form based Interface- this is the type of interface that has been used while developing
the relational database for the part details of the robot. It basically displays a form to
the user and bears various options for the user to retrieve the data, update it, delete, or
insert new data. Form based interfaces are designed for the everyday users who are
not very familiar with database concepts.
• Menu based interface- this interface helps the user to pick various options from a menu list displayed by the system. Hence, the query is composed step by step by picking options from a displayed menu list. They are often used as browsing interfaces to help the user look through the content of the database.

• Natural language interface- these accept requests written in English and then understand it and covert it to the language understood by the DBMS. Natural language interfaces usually refer to words in its schema as well as a set of standard words while trying to interpret the request.

• Parametric user interface- certain types of users make use of a set of operations repeatedly, thus the DBA design a particular interface for such users. It includes various hot keys and shortcuts to help the user implement the required option with ease.

• The DBA interface- The Database administrators have the privileges to grant and revoke rights to other users. They also have the option to change the schema and reorganize the data structure. The interface they usually use is SQL plus, as in the case of Oracle database administrators.
Chapter Two

Classification of Database Management systems-

The DBMS is mainly classified on the basis of its data model. The most commonly used DBMS systems are Relational, Hierarchical, and Network and Object Oriented \[^6\]. DBMS are also classified on the basis number of users using the system for example single user systems support only one user at a time whereas multiple user systems support a large number of users simultaneously. Another criterion for classifying the DBMS is its cost, Single user low end systems cost between $100 to $3000 whereas Multi user higher level systems cost more then$10,000. The DBMS may also be classified based on of the number of sites over which the database is distributed. Centralized databases have their data stored on one computer they have the ability to support multi users however the entire databases. Based on the classification by Data Model consider the following types of Databases.

2.1 Flat-File Databases

The simplest database form consists of one table with records having enough columns to contain all of the data you need to describe the entity class. The term flat-file \[^6\] is derived from the fact that the database itself is two-dimensional— the number of table fields determines the database's width, and the quantity of table records specifies its height. There are no related tables in the database, so the concept of data depth, the third dimension, does not apply. Any database that contains only one table is, by definition, a flat-file database if the database requires that the tables be flat. Flat-file databases are suitable for simple telephone and mailing lists. Ranges of cells, which are designated as
"databases" by spreadsheet applications, also are flat-files. A mailing-list database, as an example, has designated fields for names, addresses, and telephone numbers. Data files used in Microsoft Word's print merge operations constitute flat-file databases. The problem with these databases occurs whenever one tries to expand it. Adding new fields causes trouble when you want to print reports. It is especially difficult to format printed reports that have repeating groups.

Regardless of the deficiencies of flat-file databases, many of the early mainframe computers only offered flat-file database structures. All spreadsheet applications offer "database" cell ranges that you can sort by a variety of methods. Although spreadsheet "databases" appear to be flat, this is seldom truly the case. One of the particular problems with spreadsheet databases is that the spreadsheet data model naturally leads to inconsistencies in attribute values and repeating groups.

2.2 The Network and Hierarchical Databases

The inability of flat-file databases to efficiently deal with data that involved repeating groups of data led to the development of a variety of different database structures (called models) for mainframe computers. The first standardized and widely accepted model for mainframe databases was the network model developed by the Committee for Data System Languages (CODASYL), which also developed Common Business-Oriented Language (COBOL) to write applications that manipulate the data in CODASYL network databases. Although the CODASYL database model has its drawbacks, an extraordinary number of mainframe CODASYL databases remains in use today. CODASYL databases substitute the term record type for table, but the characteristics of a CODASYL record type are fundamentally no different from the properties of a table.
CODASYL record types contain pointers to records of other record types. A pointer is a value that specifies the location of a record in a file or in memory. For example, a customer record contains a pointer to an invoice for the customer, which in turn contains a pointer to another invoice record for the customer, and so on. The general term used to describe pointer-based record types is linked list; the pointers link the records into an organized structure called a network. Network databases offer excellent performance when you are seeking a set of records that pertain to a specific object because the relations between records (pointers) are a permanent part of the database. However, the speed of network databases degrades when you want to browse the database for records that match specific criteria. The problem with CODASYL databases is that database applications (primarily COBOL programs) need to update the data values and the pointers of records that have been added, deleted, or edited. The need to sequentially update both data and pointers adds a great deal of complexity to transaction-processing applications for CODASYL databases.

IBM developed the hierarchical model for its IMS mainframe database product line, which uses the DL/1 language. The hierarchical model deals with repeating groups by using a data structure that resembles an upside-down tree: Data in primary records constitute the branches and data in repeating groups are the leaves. The advantage of the hierarchical model is that the methods required finding related records are simpler than the techniques needed by the network model. As with the CODASYL model, there are a large number of hierarchical databases running on mainframe computers today.
2.3 The Relational Database Model

The relational database model revolutionized the database world and enabled PCs to replace expensive minicomputers and mainframes for many database applications.

Dr. E.F. Codd of IBM’s San Jose Research Laboratories developed the relational database model in 1970. The primary advantage of the relational model is that there is no need to mix pointers and data in tables. Instead, relations between attribute values link records. A relation consists of a linkage between records in two tables that have identical attribute values. Consider an example as shown below where we have the following data: Name, Customer number, Ship to address, Inventory No. and Prod No. and Quantity. This example will be used to explain further concepts that follow. The figure below illustrates relations between attribute values of relational tables that constitute part of a sales database.

![Figure 4- Relationships between a table](image)

Figure 4- Relationships between a table
Because relational tables do not contain pointers, the data in relational tables is independent of the methods used by the database management system to manipulate the record’s \[4\]. A relational database management system is an executable application that can store data in and retrieve data from sets of related tables in a database. The RDBMS creates transitory virtual pointers to records of relational tables in memory. Virtual pointers appear as they are needed to relate (join) tables and are disposed of when the relation is no longer required by a database application. The "joins" between tables are shown as dashed lines in the figure above. Joins are created between primary key fields and foreign key fields of relational tables. The concepts of foreign keys and primary keys are explained in detail in the next section of this thesis. Considering the same example as above, the primary and foreign key fields of the tables of Figure-A are listed in the table below.

The primary and foreign keys of the tables depicted in Figure 4

<table>
<thead>
<tr>
<th>Table</th>
<th>Primary Key</th>
<th>Foreign Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customers</td>
<td>Cust#</td>
<td>None</td>
</tr>
<tr>
<td>Invoices</td>
<td>Inv#</td>
<td>Cust#</td>
</tr>
<tr>
<td>Invoice Items</td>
<td>Inv# and Prod#</td>
<td>Inv#</td>
</tr>
</tbody>
</table>

Table A- Primary Key and Foreign keys in a Table.

Relational databases require duplicate data among tables but don't permit duplication of data within tables. It is required to duplicate the values of the primary key of one table as the foreign key of dependent table’s \[6\]. A dependent table is a table that requires a relationship with another table to identify its entities fully. Dependent tables often are
called secondary or related tables. Thus, the Invoices table is dependent on the Customers table to supply the real-world name and address of the customer represented by values in the Cust# field. Similarly, the Invoice Items table is dependent on the Invoices table to identify the real-world object (in this case an invoice) to which records are related.

There are three types of relations defined by the relational database models, each of which is described in the following list:

*One-to-one* relations require that one and only one record in a dependent table can relate to a record in a primary table[^10]. One-to-one relations are relatively uncommon in relational databases.

*One-to-many* relations enable more than one dependent table to relate to a record in a primary table. The term *many-to-one* is also used to describe one-to-many relations. One-to-many relations constitute the relational database model's answer to the repeating-groups problem. Repeating groups are converted to individual records in the table on the "many" side of the relation. One-to-many relations are the most commonly found relations.

*Many-to-many* relations are not true relations, because many-to-many relations between two tables require an intervening table, called a relation table, to hold the values of the foreign keys.

A set of steps, discussed in a forthcoming chapter, define the process of creating the physical schema that conforms to the relational model.
Chapter Three

Figure 5. PHASES OF DATABASE DESIGN
3.1 PHASES OF DATABASE DESIGN - The above diagram shows the generalized process involved in creating a DBMS. It consists of mainly two sections. The first one, which is DBMS independent, involves collecting requirements, designing a conceptual schema, and a generalized functional analysis \[10\]. The other half is DBMS specific and involves working on the logical schema and then actually converting that into a physical schema. It also involves designing the program interface. Let us look at the main steps one must follow when it comes to designing the database

3.1 a Model the Data

The first step in designing a relational database is to determine what objects need to be represented by database entities and what properties of each of these objects require inclusion as attribute classes. The process of identifying the tables required in the database and the fields that each table needs is called data modeling \[21\]. There are two approaches that you can take during the process of data modeling:

*Application-oriented design* techniques start with a description of the type of applications required by the potential users of the database. From the description of the application, you design a database that provides the necessary data. This is called the *bottom-up* approach, because applications are ordinarily at the bottom of the database hierarchy.

*Subject-oriented design* methodology begins by defining the objects that relate to the subject matter of the database as a whole. This approach is called *top-down* database design. The content of the database determines what information front-end applications can present to the user.

Although application-oriented design can enable you to quickly create an *ad hoc* database structure and the applications to accomplish a specific goal, bottom-up design is
seldom a satisfactory long-term solution to an organization's information needs. It is common to find several application-oriented databases within an organization that have duplicate data, such as independent customer lists. When the firm acquires a new customer, each of the customer tables needs to be updated. This is an inefficient and error-prone process.

Subject-oriented database design is a far more satisfactory method. You might want to divide the design process into department-level or workgroup-related databases, such as those in the following list:

A *sales* database that has tables based on customer, order and line item, invoice and line item, and product entity classes.

A *production* database with tables for parts, suppliers, bills of material, and cost accounting information. The product and invoice tables of the sales department's database would be attached to the production database.

A *personnel* database with tables for employees, payroll data, benefits, training, and other subjects relating to human resources management. The production and sales databases would attach to the employee’s table—production for the purposes of cost accounting purposes, and sales for commissions. An *accounting* database, with tables comprising the general ledger and subsidiary ledgers, would attach to the majority of the tables in the other databases to obtain access to current finance-related information. Accounting databases often are broken down into individual orders, accounts receivable, and accounts payable and general ledger databases.
3.1b **Draw Database Diagrams** - Diagramming relations between tables can aid in visualizing database design. *Entity-relation* (E-R) diagrams, also called *entity-attribute-relation* (EAR) diagrams, are one of the most widely used methods of depicting the relations between database tables. Peter Chen introduced the E-R diagramming method in 1976. An E-R diagram consists of rectangles that represent the entity classes (tables). Ellipses above the table rectangles show the attribute class (field) involved in the relation. Pairs of table rectangles and field ellipses are connected by parallelograms to represent the relation between the fields.

The figure below illustrates an E-R diagram for the Customers and Invoices tables of the database described in Figure A. The 1 and m adjacent to the table rectangles indicate a one-to-many relationship between the two tables. E-R diagrams describe relations by predicates. One of the definitions of *predicate* is "a term designating a property or relation." E-R diagrams are capable of describing virtually any type of allowable relation between two tables by adding additional symbols to the basic diagram shown in the above figure. A very large number of E-R diagrams are required to define relationships between the numerous entities in enterprise-wide databases.

![ER Diagram](image)

Figure 8. ER Diagram
3.1 c Step Three

C) Normalize the Table Data- The process of transforming existing data into relational form is called normalization \(^{[22]}\). Normalization of data is based on the assumption that you have organized your data into a tabular structure wherein the tables contain only a single entity class. The objectives of data normalization are:

- To eliminate duplicated information contained in tables.
- To accommodate future changes to the structure of tables.
- To minimize the impact of changes to database structure on the front-end applications processing the data.
- To isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.

In Relational database design, normalization usually involves dividing a database into two or more tables and defining relationships between the table’s \(^{[14]}\). There are three main normal forms, each with increasing levels of normalization:\(^{[8]}\).

- First Normal Form (1NF): Each field in a table contains different information. For example, in an employee list, each table would contain only one birthday field.
- Second Normal Form (2NF): No field values can be derived from another field. For example, if a table already included a birthdate field, it could not also include a birth year field, since this information would be redundant.
- Third Normal Form (3FN): No duplicate information is permitted. So, for example, if two tables both require a birthdate field, the birthdate information would be separated into a separate table, and the two other tables would then
access the birthdate information via an index field in the birthdate table. Any change to a birthdate would automatically be reflected in all tables that link to the birthdate table.

There are additional normalization levels, such as Boyce Codd Normal Form (BCNF), fourth normal form (4NF) and fifth normal form (5NF). While normalization makes databases more efficient to maintain, they can also make them more complex because data is separated into many different tables.

3.1d Step Four

D) Create a Primary Key for Each Table - It is important to create a unique numerical key in the first field of most tables created. This field is called a primary key. This field will usually be both an index and the first field of the database. It is the primary key for the table, and must be, by definition, unique. That is, each record should have a unique Code field associated with it.

The primary key

- Serves as the means of differentiating one record from another
- Provides Referential integrity
- Help with fast searches and sorts

Considering the example:

**Right Method**

**CustNo**: Integer

LastName, FirstName, Address, City, State, Zip: string

**Wrong Method**

LastName, FirstName, Address, City, State, and Zip: string
The first example is "correct" because it has a primary index called CustNo. It is declared as a unique Integer value. The second example is "wrong" because it omits a simple numerical field as the primary index.

The important point to remember is that these fields allow the database to be treated as nothing more than sets of simple integers related together in various combinations. Adding these extra index fields to your tables makes the data computer-friendly.

One-to-Many Relationships: The Data and the Index

Consider the Customer, Orders, Items, and Parts tables from a database. All four of these tables are related in one-to-many relationships, each-to-each. That is, the Customer table is related to the Orders table, the Orders table to the Items table, and the Items table to the Parts table.

<table>
<thead>
<tr>
<th>Master</th>
<th>Detail</th>
<th>Connector (primary key and foreign key)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Orders</td>
<td>CustNo</td>
</tr>
<tr>
<td>Orders</td>
<td>Items</td>
<td>OrderNo</td>
</tr>
<tr>
<td>Items</td>
<td>Parts</td>
<td>PartNo</td>
</tr>
</tbody>
</table>

Table B- Relationships and connections across tables.

The preceding list shows that the Customer and Orders tables are related in a one-to-many relationship, with Customer being the master table and Orders being the detail table. The connector between them is the CustNo field. That is, they both have a CustNo field. The CustNo field is the primary key of the Customer table and the
foreign key of the Orders table. The OrderNo field is the primary key of the Orders table and a foreign key of the Items table. The PartNo field is the primary key of the Parts table and a foreign key of the Items table.

The relationship between these tables can be reversed. For instance, the Parts table could become the master table and the Items table the detail table, and so on, back down the line. The Customer table has a series of CustNo fields. To get the Orders associated with that customer, you ask this question: "What are all the rows from the Orders table that have a CustNo of 50?" That is:

Select * from Orders where CustNo = 50

Clearly, one could reverse this question. If you select a particular row from the Orders table, you could find which item from the Customer table it is related to by asking for the set of all Customer records with a CustNo of 50. Because the CustNo field for the Customer table is a unique index, you will get only one record back. However, the way you relate the tables is still the same:

Select * from Customer where CustNo = 50

The Parts, Orders, Items, and Customer tables have various keys. As it happens, these keys are also indexes. An index enables us to sort tables on a particular field. A key helps us define the relationship between two tables, or otherwise group related bits of information by a set of predefined and automatically enforced rules.

However one can still relate tables even without the presence of any keys or indexes. For example, if there were no CustNo primary and foreign keys in the Customer and Orders tables, one could still use SQL to relate the tables in a one-to-many relationship. However, in this scenario, performance would be slow because there is no index, and there would be no constraints on the data you could enter in the two tables because there
would be no primary and foreign keys that define referential integrity. Which would result in what we call “Junk Data”. Now consider the 2 keys the first kind is called a primary key while the second is called a foreign key.

- A primary key is a unique value used to identify a record in a table \(^6\). It is usually numerical, and it is usually indexed. It can be combined with a foreign key to define referential integrity.

- Since it is indexed, the primary key defines the default sort order for the table. When you first open up a table, it will be automatically sorted on this field. If a table does not have a primary index, records will appear in the order in which they were added to the table.

- The primary index must be unique. Which means that there can’t be two CustNos in the Customer table that is the same. However there can be multiple foreign keys that are not unique.

- The primary and foreign keys are never composite. They always consist of one field.

Creating a primary key enables one to have two people with the same name, but with different addresses. For instance, one can list a John Doe on Maple Street who has a CustNo of 25, and a John Doe on Henry Street who has a CustNo of 2000. The names may be the same, but the database can distinguish them by their CustNo. If the database had to sort on the address fields every time it tried to distinguish these two John Does, it would take a long time for the sort to finish.
Secondary Indices and Foreign Keys

The CustNo field of the Orders table is a foreign key because it relates the Orders table to the primary key of the Customer table. It is also a secondary index, which aids in sorting and searching through data. Indices also speed up operations such as joins and other master-detail relationships. Here are some facts about foreign keys and secondary indexes.

A foreign key provides a means for relating two tables according to a set of predefined rules called referential integrity.

- Using SQL one can relate two tables in a one-to-many relationship even if there is no index or key in either table. However, the performance improves with indexes.
- It is impossible to relate two tables in a one-to-many relationship without indexes.
- A secondary index provides an alternative sort order to the one provided by the primary key.
- It is needed to explicitly change the index in order to switch from the primary index to a secondary index.
- An index that contains more than one field are called a composite index. Composite secondary indexes, can also be created which means the indexes will contain multiple fields. In practice, fields such as FirstName and LastName are often part of a secondary index, because the primary index is usually a unique numerical value. Sometimes a primary index will consist of three fields, such as the CustNo, FirstName, and LastName fields.
Chapter Four

The Language and the DBMS.

4.1 WHY SQL?

SQL began in IBM laboratories in San Jose California, where it was developed in the late 1970’s. SQL also known as Structured query language was originally developed for IBM’s DB2 products; it is SQL, which makes the RDBMS possible.

During 1984 everyone was talking about SQL. In brief, every vendor said “IBM will make a big deal of DB 2 and SQL. I want to be compatible with IBM.” A similar message was conveyed by so-called value-added resellers (VARS) who said “I want application code that I write to run both on your data manager and on DB 2”. Discussions with the vendors concerning exactly what they meant by SQL and exactly what they wanted in terms of compatibility usually evoked an answer of “I don’t know”. People were not exactly sure of what they wanted. Eventually, representatives of large users of database services picked up this message. The message that they picked up was: “I need to run my applications on the IBM system and on the systems of X, Y, and Z. I plan to move to DB 2 and I want to ensure that the DB 2 applications I write, can be moved to the systems of other vendors. SQL is the mechanism that will allow me to achieve this objective.” Consequently, all vendors of data base systems put in place plans to support SQL. Moreover, all other query languages (e.g. QUEL, Datatrieve, etc.), regardless of their intellectual appeal, became “sunset” interfaces, i.e. they began to slowly fade away and become a thing of the past.
The greatest advantage of SQL is the fact that it is user friendly unlike the third generation languages (3GLs) such as COBOL and C.

SQL helps the user achieve the following on an RDBMS

- Modify a database structure
- Change system security settings
- Add additional user permissions on tables
- Query a database to obtain required information
- Update the database

To achieve the above objectives SQL uses various statements; these can be broadly classified into the following

4.2 **DDL - Data definition Language**[^27]: DDL allows the user to

1) Create a database table
2) Drop a database table
3) Alter a database object
4) Revoke privileges on a database object
5) Grant privileges on a database object

The table below highlights the various DDL statements and their output.

<table>
<thead>
<tr>
<th>SQL Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alter Procedure</td>
<td>Alters a stored procedure</td>
</tr>
<tr>
<td>Alter Table</td>
<td>Adds columns, redefines columns, changes storage location.</td>
</tr>
</tbody>
</table>
Analyze  Gathers performance statistics for the database objects.

Alter table add constraint  Adds a constraint to the existing table.

Create Table  Creates a table.

Create Index  Creates an Index.

Drop Index  Drop’s an index.

Drop Table  Drop’s a table.

Grant  Grants user privileges.

Truncate  Deletes all rows from table.

Revoke  Removes the privileges granted earlier.

4.3 DML - Data Manipulation Language[27]: The data manipulation language is the one used by most of the users accessing the database. The DML allows the user to update, insert, delete and select data in the database. The DML statements allow the user to manipulate the data in the database according to the requirement. The table below highlights the various DML statements and their output.

<table>
<thead>
<tr>
<th>SQL Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>Add rows of data to the table.</td>
</tr>
<tr>
<td>Delete</td>
<td>Deletes rows from table.</td>
</tr>
<tr>
<td>Select</td>
<td>Retrieves rows of data from table.</td>
</tr>
<tr>
<td>Update</td>
<td>Changes data in table.</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Commit</td>
<td>Writes data to disk.</td>
</tr>
<tr>
<td>Rollback</td>
<td>Undo the changes before last Commit.</td>
</tr>
</tbody>
</table>

Hence SQL was the way to go to ensure a DBMS which would withstand the constant change in technology and not be outdated in a month, now the next main step was to obtain a user friendly GUI accessible DBMS which would communicate with SQL effectively.

The DBMS System – There were a lot of different database packages out there, but what was needed was something user friendly, something on the cutting edge of technology, which would not die out with time and something compatible with new technologies as well as IBM friendly. Oracle 7.0 offered all the above features also they have been pretty much the leaders in the DBMS field. The software offered various advantages

4.4 Why Oracle?

With the release of Personal Oracle7 for Windows 95, Oracle has almost completely revamped the look and feel of the product: A tool called the Navigator integrates the capabilities of the previous products. In place of Database Manager, two new programs--; Start Database and Stop Database--; simplify the operation of Personal Oracle7. The Navigator can be invoked by going to the PersonalOracle7 for Windows 95 program group and selecting Personal Oracle7 Navigator. The Navigator on being invoked, opens a window with two frames--Two folders--; Projects and Database Connections--; and a
A database system is the key to solving the problems of information management. The Oracle database provides efficient and effective solutions with the following features:

- **Client/server (distributed processing) environments** - To take full advantage of a given computer system or network, Oracle allows processing to be split between the database server and the client application programs. The computer running the database management system handles all of the database server responsibilities while the workstations running the database application concentrate on the interpretation and display of data.

- **Large Database sizes** - Oracle supports the largest of databases, potentially terabytes in size. To make efficient use of expensive hardware devices, it allows full control of space usage.

![Navigator Interface](image)

**Figure 7. Oracle Navigator Interface**

cylindrical icon identified as Local Database appear within the left frame of the main window.
Many concurrent database users - Oracle supports large numbers of concurrent users executing a variety of database applications operating on the same data. It minimizes data contention and guarantees data concurrency.

High Transaction processing performance - Oracle maintains the preceding features with a high degree of overall system performance. Database users do not suffer from slow processing performance. At some sites, Oracle works 24 hours per day with no down time to limit database throughput. Normal system operations such as database backup and partial computer system failures do not interrupt database use.

Controlled Availability - Oracle can selectively control the availability of data, at the database level and sub-database level. For example, an administrator can disallow use of a specific application so that the application’s data can be reloaded, without affecting other applications.

Meets Industry Standards - Oracle adheres to industry accepted standards for the data access language, operating systems, user interfaces, and network communication protocols. It is an “open” system that protects a customer’s investment. Also, Oracle7 has been evaluated by the U.S. Government’s National Computer Security Center (NCSC) as compliant with the Orange Book security criteria; the Oracle also supports the Simple Network Management Protocol (SNMP) standard for system management. This protocol allows administrators to manage heterogeneous systems with a single administration interface.

Manageable security - To protect against unauthorized database access and use, Oracle provides fail-safe security features to limit and monitor data access. These features make it easy to manage even the most complex design for data access.
- Data Integrity - Oracle enforces data integrity, “business rules” that dictate the standards for acceptable data. As a result, the costs of coding and managing checks in many database applications are eliminated.

- Distributed Systems - For networked, distributed environments, Oracle combines the data physically located on different computers into one logical database that can be accessed by all network users. Distributed systems have the same degree of user transparency and data consistency as non—distributed systems, yet receive the advantages of local database management.

Due these offered advantages Oracle 7 was used as the Database system. The end user is not expected to be familiar with the concepts of Database design and SQL coding and hence it called for designing a user friendly end user interface for the end user. The most feasible option was using a graphic user interface (GUI) which would essentially consist of forms which would prompt the end user to input the part number and the backend triggers would then access the database and provide the end user with the necessary information. The GUI was designed using Developer 2000 Rel. 2.0, various user-friendly forms help access the required information from the database.
Chapter Five
The Building of the database

5.1 The ER diagram (Logical):

![ER Diagram](image_url)

Figure 9. ER Diagram (Logical)
The ER Diagram was created as above. This has to be converted this into a physical schema. The database tables had to be created and then populated. Using SQL the scripts were written and then executed to create tables.

Figure 10. SQL Image - Tables Created.

The figure above shows the tables that had been created. And the figure below shows the structure of those tables. As we can see that the Part_No and the Vendor Name were the Primary keys.
Once the tables had been created and populated, the next step was to design the program interface. Using Developer 2000\textsuperscript{[19]} the Program interface was designed. The process involved creating various forms and using PL/SQL\textsuperscript{[18]} to execute the SQL in the back end upon the interaction by the end user on the GUI. These scripts are attached to the appendix.

Below are some screen shot of the GUI Interface. Using the GUI the end user can easily insert new records. This option can be disabled and set to make sure that only the administrators have access to write to the tables.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{Fig11}
\caption{Datatypes used.}
\end{figure}
The next screen displays how the record inserted can be saved and hence updated in the tables on the database.

Figure 12. GUI Interface (Insert Option)
The screen above shows the flexibility the GUI offers as to saving record to the Database.

The Detailed SQL scripts for the project have been attached in the Appendix.

5.2 Improvements & Test Analysis-

Improvements-The DBMS was presented to the robot team in order to obtain feedback from the team and the enhancement’s desired. Based on the feedback the following enhancement requests were obtained.

a) Ability to use Microsoft Access to browse through the data.

b) Ability to easily provide soft copies of the database to other end users.
c) Some parts had different vendors for repair purposes. This information was needed as well.

d) On the order form a functionality so as to provide an option to enter the account used for the purchase order was needed. This would help in keeping track of the payments being made.

e) Easy access to the database over the web.

f) Basic training for updating and inserting information into the database system.

g) Displaying Usage or functionality of each part.

The database was also extracted into Microsoft Access\textsuperscript{[26]} 97. The tables were now available in the access database. However there was the need to create a user friendly interface, two basic forms were created namely Part information and Vendor Information, the latter provided the vendor information, whereas the prior form gave details for the part and its dimensions. The access database also had a report designed to provide a report printout containing the Part number, Part name and Vendor name.

The database had to be rearchitected so as to meet the new recommendations. The modified ER Diagram is as shown below.
** Items in Blue indicate the added components to the database schema.

Figure 14. Modified ER Diagram

Testing:

Using a systematic approach the database was put to test. It was required to perform a quality analysis and make any changes/improvements as required. Two groups consisting of four test subjects each were used initially to perform stress analysis for the database.
Group A-
This group was presented with an form indicating the part number which had to be ordered. The group was timed, to obtain the measure of time it took them to actually reach the correct answer. The purpose of the study was to ensure that the application was easy to use and intuitive.

<table>
<thead>
<tr>
<th>Test Subject</th>
<th>Part Battery</th>
<th>Part =1515L-1</th>
<th>Part=SW10</th>
<th>Total Time Taken(Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>✓</td>
<td>*</td>
<td>✓</td>
<td>228</td>
</tr>
<tr>
<td>B</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>232</td>
</tr>
<tr>
<td>C</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>212</td>
</tr>
<tr>
<td>D</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>198</td>
</tr>
</tbody>
</table>

Table C –Test Analysis for group A.

* - This end user was unable to get the required information since he did not realize that the part number was case sensitive. To avoid such errors from occurring in the near future, the GUI has been modified so as to directly convert the part number entered into uppercase. This has resolved the problems the end users may face due to the case sensitivity.

Group B-
This group was presented with parts that needed repair. The results for the study have been tabulated below.
<table>
<thead>
<tr>
<th>Test Subject</th>
<th>Part Battery</th>
<th>Part =1515L-1</th>
<th>Part=SW10</th>
<th>Total Time Taken(seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>246</td>
</tr>
<tr>
<td>B</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>189</td>
</tr>
<tr>
<td>C</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>195</td>
</tr>
<tr>
<td>D</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>226</td>
</tr>
</tbody>
</table>

Table D- Test Analysis for Group B

The tests above helped in better understanding of the end users point of view and helped in narrowing down probability of an error being, generated.

5.3 Where do we go from here?

The possibilities and further scope of this project is endless. This thesis was meant to provide some insight into those future advances. We have seen the OLTP i.e. the transaction system side of databases, however we also need to explore the OLAP i.e. Online analytical processing side of the Database technology. Database systems can also serve the purpose of a Warehouse as such and help in storing large data volumes. Corporations have a variety of on-line transaction processing (OLTP) systems such as financial, order entry, work scheduling, and point-of-sale systems which create operational data. This data is part of the corporate infrastructure and is detailed, non-redundant, updateable and reflects current period values. In contrast, data required by decision support analysts is often summarized, has a lengthy time horizon, is redundant to the support of varying data views, and is non-updateable. In order to provide data to decision support analysts, relevant operational data is extracted from OLTP systems,
cleansed, encoded, and summarized. After being transformed into a format suitable for decision support, the data is uploaded into the data warehouse.

- The DSS (decision support system)\(^7\) engine is the heart of the DSS architecture. It transforms data requests into SQL queries to be sent to the data warehouse and formats query results for presentation to the DSS analyst. To support these functions, the DSS engine includes a dynamic SQL query generator, a multidimensional data analysis engine, a mathematical equation processor, and a cross-tabulation engine\(^7\). The Application Development Environment (ADE) consists of a suite of object-oriented tools for the construction of the application used by DSS analysts. The DSS application allows, at a minimum, the specification of data to be retrieved and the viewing of query results.

Additionally, the Application Development Environment should permit:

- Development of DSS applications that automate white-collar work. Tasks such as writing letters or sending electronic mail messages, performing sophisticated mathematical analyses, or issuing requests for more detailed information often directly follow decision support investigations and should be automated activities.

- Development of closed-loop Decision Support Systems. Traditional DSS systems

![Figure 15 - Closed loop decision support](image-url)
are "open-loop" in that they present data to DSS analysts and then require some action to be taken. Closed-loop systems interpret query results and automatically execute transactions such as changing product prices or shifting product inventory. At a minimum, these systems should allow the analyst to create operational transactions while interactively viewing DSS query results, as depicted in Figure 14.

Since decision support activity is typically not complete after data retrieval, strong data delivery architectures have tight integration with desktop personal productivity applications such as word processors, spreadsheets, electronic mail, and personal database managers. These linkages allow data to flow directly into presentation, analytical, and communication tools already used by the DSS analyst. Query results are directed to these applications using industry-standard data interfaces such as DDE, OLE, MAPI, and OD.

Many corporations have completed and operationalized their initial executive information and decision support applications with moderate success. Users have recognized the value of Decision Support Systems and are driving the demand for applications that span additional business functions and are deeper in scope than initial pilot applications. Information system developers have used a variety of tools to create these Decision Support Systems, including 3rd generation languages, 4GL application development tools, and Executive Information System tools.
Chapter Six

Conclusion

We have just barely scratched the surface of the database technology; the possibilities that lie ahead are endless. This same system can eventually serve to be a data warehouse in the longer run. For example, to find out what parts of the inventory need to be reordered this is called data mining: a hot buzzword for a class of database applications that look for hidden patterns in a group of data. The term is commonly misused, describing data mining as software that presents data in new ways. True data mining software doesn't just change the presentation, but actually discovers previously unknown relationships among the data.

Consider a production plant which stores all its inventory information on a data warehouse which eventually gets populated at the end of the day with the collected OLTP data. We have tools which will sort through that limitless data and figure out what parts of the inventory need to be reordered, and not simply stop at this stage but go one step further by comparing the prices being offered by all the current vendors supplying that part. Next, the tool places an order with the most efficient vendor depending on the criticality of the situation. By criticality I mean, is it more important to consider the cost factor or the time factor. Eventually what we are looking at is a fully operational intelligence to ensure that the company does not run out of Inventory supply.

The database has been, successfully implemented for the robot team. The benefits are already visible, the end user can now fetch all relevant information for the robot parts using the database created. Hence even though the migration from the orthodox paper
pencil and register record keeping may seem painful initially, but by doing so we only stand to gain.
References

25. Oracle is the registered Trademark of the Oracle Corp. http://www.oracle.com
26. MS Access is the registered Trademark of Microsoft Inc. http://www.microsoft.com
CREATE SCRIPTS FOR THE FOUR TABLES NAMELY

PART:
create table part(part_no varchar2(25) primary key, part varchar2(25),
length varchar2(10), breadth varchar2(10), height varchar2(10), regular varchar2 (1));

ORDERFORM:
create table orderform(part_no varchar2(25) primary key, vname varchar2(25), price
number(15), pieces number(10));

VENDOR;
create table vendor22(part_no varchar2(25) not null, vname varchar2(25) not null, address
varchar2(45), phone varchar2(12), price float(126), pin number (5), constraint aa foreign
key (part_no) references part(part_no));

REPAIR_VENDOR;
create table repair_vendor as select * from vendor;

POPULATING SCRIPTS FOR TABLE VENDOR:

insert into vendor values
('1515L-1', 'Volker Controls', '590 Congress park drive, centreville-Oh', '9374338128', '20.33',
45459);

insert into vendor values
('1515L-2', 'Volker Controls', '590 Congress park drive, centreville-Oh', '9374338128',
'13.90', 45459);

insert into vendor values
('1515L-3', 'Volker Controls', '590 Congress park drive, centreville-Oh', '9374338128',
'11.51', 45459);
insert into vendor values
('1515L-4','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'9.41', 45459);

insert into vendor values
('1515L-5','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'6.47', 45459);

insert into vendor values
('1515L-6','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'6.05',45459);

SQL> insert into vendor values
  2  ('1010L','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
  3  '3.74',45459);

SQL> insert into vendor values
  2  ('3320','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
  3  '8.25',45459);

insert into vendor values
('4350','Volker Controls','590 Congress park drive, centreville-Oh','9374338128',
'5.30',45459);

insert into vendor values
('4351','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'6.75',45459);
insert into vendor values
('3280','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'0.57',45459);

insert into vendor values
('4101','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'3.90',45459);

insert into vendor values
('4302','Volker Controls','590 Congress park drive,centreville-Oh','9374338128',
'2.80',45459);

insert into vendor values
('DC48A','J.N.Fauver Company','11253 Williamson Rd,Cincinnati-Oh','5132479900',
'300.00','45241');

insert into vendor values
('0798-39-003','J.N.Fauver Company','11253 Williamson Rd,Cincinnati-Oh','5132479900',
'900.00','45241');

insert into vendor values
('ISCAN','ISCAN Corp','2370 Dixie Hwy,Hamilton-Oh','5138681766',
'9000.00','45015');

insert into vendor values
('JVC22','Bestbuy','865 East Kemper Rd.,Springdale-Oh','5136714305',
'400.00','45245');

insert into vendor values
insert into vendor values
('MAXIM442', 'Electro', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', '20.00', '45015');

insert into vendor values
('PS112', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', '50.00', '45015');

insert into vendor values
('DMC1030', 'Galil Motion Control Inc', '575 Maude Court, Sunnyvale-Ca', '8003776329', '900.00', '94086');

insert into vendor values
('ICM1100', 'Galil Motion Control Inc', '575 Maude Court, Sunnyvale-Ca', '8003776329', '150.00', '94086');

insert into vendor values
('COMPUTER', 'UC Bookstore', 'University of Cincinnati-Oh', '5135561700', '1200.00', '45219');

insert into vendor values
('RS232', 'J.N.Fauver Company', '11253 Williamson Rd, Cincinnati-Oh', '5132479900', '100.00', '45241');

insert into vendor values
('BATTERY', 'Michel tire & Co', 'Fairfield, 6500 Dixie Hwy, Oh', '5138748808', '64.50', '45014');

insert into vendor values
('BATTERY', 'Yuasa Excide Inc', 'Po Box 14145, Reading, PA', '2152504657', '68.00', '19612');

insert into vendor values
(‘INVERTER’,‘Reliance Electric’,’6950 Washington avenue,Minnesota’,’6129423600’,’600.00’,’55344’);

insert into vendor values
(‘MME1’,‘J.N.Fauver Company’,’11253 Williamson Rd,Cincinnati-Oh’,’5132479900’,’651.90’,’45214’);
insert into vendor values
(‘GEARBOX’,‘Boston Gearboxes’,’7970 Hwy 25,Florence,Ky’,’6015258021’,’340’,’41042’);

insert into vendor values
(‘P2BSCM100’,‘Cincinnati Belting’,’737 W6,Queensgate,Oh’,’5136219050’,’31.56’,’45203’);

insert into vendor values
(‘CPS4’,‘Grainger Inc’,’4420 Glendale,Milford Rd Oh’,’5135637100’,’6.00’,’45013’);

insert into vendor values
(‘SK4’,‘Grainger Inc’,’4420 Glendale,Milford Rd Oh’,’5135637100’,’4.28’,’45013’);

insert into vendor values
(‘90-SERIES 6S’,‘Borne’,’1526 Scott St,Covington Ky’,’6012915447’,’27.75’,’41012’);

insert into vendor values
(‘DR2PN’,‘Borne’,’1526 Scott St,Covington Ky’,’6012915447’,’30.00’,’41012’);

insert into vendor values
(‘CB8’,‘Michel tire & Co’,’Fairfield,6500 Dixie Hwy,Oh’,’5138748808’,’2.00’,’45014’);

insert into vendor values
(‘EC10’,‘Michel tire ’,’Fairfield,6500 Dixie Hwy,Oh’,’5138748808’,’1.50’,’45014’);
insert into vendor values
('SW10', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 1.50, '45015');

insert into vendor values
('CN30', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 0.75, '45015');

insert into vendor values
('10SWG', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 15.00, '45015');

insert into vendor values
('16SWG', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 20.00, '45015');

insert into vendor values
('RSW1', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 35.00, '45015');

insert into vendor values
('JSK1', 'Radioshack', '2370 Dixie Hwy, Hamilton-Oh', '5138681766', 30.00, '45015');

insert into vendor values
('PXGLASS', 'Cincinnati Plastics', '4560 Cornell Rd Blue Ash, Oh', '5132479449', 60.00, '45012');

insert into vendor values
('FSTN84', 'Officemax', '10166 Colerain Av, Oh', '5133855409', 0.70, '45254');

insert into vendor values
('KEELS', 'Cincinnati Plastics', '4560 Cornell Rd Blue Ash, Oh', '5132479449', 0.60, '45012');
insert into vendor values
('RLYSW1','Auto Zone','3146 Reading rd Avondale,Oh','5139610700',24.00,'45219');
insert into vendor values
('PPKTS2','Bestbuy','865 East Kemper Rd.,Springdale-Oh','5136714305',295.00,'45245');
insert into vendor values
('PS1','Auto Zone','3146 Reading rd Avondale,Oh','5139610700',90.00,'45219');

OTHER TABLES POPULATED USING DEVELOPER 2000;
PL/SQL code for trigger- When Validate item- TO ENSURE DATA INTEGRITY:

Begin default enforce data integrity constraint VENDOR_FK section
--
declare
cursor primary_cur is select 'x' from SCOTT.PART
   where PART_NO = :VENDOR.PART_NO;

   primary_dummy char(1);
begin
   if ( ( :VENDOR.PART_NO is not null ) ) then
      open primary_cur;
      fetch primary_cur into primary_dummy;
      if ( not primary_cur%found ) then
         message('Foreign key value does not currently exist in the primary key table.');
         close primary_cur;
         raise form_trigger_failure;
      end if;
      close primary_cur;
   end if;
end;
--
-- End default enforce data integrity constraint VENDOR_FK section
--

PL/SQL FOR TRIGGER WHEN BUTTON PRESSED- SHOW VENDOR
declare
    --test1 varchar2(25);
begin
    select PART_NO into :GLOBAL.chirag from PART where PART_NO = :PART_NO;
    open_form('VENDOR');
    --:PART_NO := test1;
end;

PL/SQL FOR TRIGGER WHEN BUTTON PRESSED- GET DETAILS (form =Part)

select part,length,breadth,height,material,regular into
    :part,:length,:breadth,:height,:material,:regular from part
where part_no=:part_no;

PL/SQL FOR TRIGGER WHEN BUTTON PRESSED- PLACE ORDER
BEGIN
select PART_NO into :GLOBAL.chirag from VENDOR where PART_NO = 
    :PART_NO;
open_form('ORDER');
END;

PL/SQL FOR TRIGGER WHEN BUTTON PRESSED- GET DETAILS (form =Vendor)
Begin
    --:PART_NO:=:GLOBAL.chirag;
    SELECT ADDRESS,PHONE,PRICE,PIN INTO :ADDRESS,:PHONE,:PRICE,:PIN
FROM VENDOR WHERE VNAME=:VNAME AND PART_NO=:PART_NO;
END;

SQL DISPLAYING DATA TYPES USED FOR TABLES;

SQL> select * from tab;
<table>
<thead>
<tr>
<th>TNAME</th>
<th>TABTYPE</th>
<th>CLUSTERID</th>
</tr>
</thead>
<tbody>
<tr>
<td>BONUS</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>DD</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>DEPT</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>EMP</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>ORDERFORM</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>PART</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>REPAIR_VENDOR</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>SALGRADE</td>
<td>TABLE</td>
<td>----------</td>
</tr>
<tr>
<td>VENDOR</td>
<td>TABLE</td>
<td>----------</td>
</tr>
</tbody>
</table>

9 rows selected.

SQL> desc part
Name desc part
<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART_NO</td>
<td>NOT NULL</td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>PART</td>
<td></td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>LENGTH</td>
<td></td>
<td>VARCHAR2(10)</td>
</tr>
<tr>
<td>BREADTH</td>
<td></td>
<td>VARCHAR2(10)</td>
</tr>
<tr>
<td>HEIGHT</td>
<td></td>
<td>VARCHAR2(10)</td>
</tr>
<tr>
<td>REGULAR</td>
<td></td>
<td>VARCHAR2(1)</td>
</tr>
<tr>
<td>MATERIAL</td>
<td></td>
<td>VARCHAR2(15)</td>
</tr>
</tbody>
</table>

SQL> desc vendor
Name desc vendor
<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART_NO</td>
<td>NOT NULL</td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>VNAME</td>
<td></td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>ADDRESS</td>
<td></td>
<td>VARCHAR2(45)</td>
</tr>
<tr>
<td>PHONE</td>
<td></td>
<td>VARCHAR2(12)</td>
</tr>
<tr>
<td>PRICE</td>
<td></td>
<td>FLOAT(126)</td>
</tr>
<tr>
<td>PIN</td>
<td></td>
<td>NUMBER(5)</td>
</tr>
</tbody>
</table>

SQL> desc orderform;
Name desc orderform;
<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART_NO</td>
<td>NOT NULL</td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>VNAME</td>
<td></td>
<td>VARCHAR2(25)</td>
</tr>
<tr>
<td>PRICE</td>
<td></td>
<td>NUMBER(15)</td>
</tr>
<tr>
<td>PIECES</td>
<td></td>
<td>NUMBER(10)</td>
</tr>
</tbody>
</table>

SQL> desc repair_vendor;
Name desc repair_vendor;
| Name | Null? | Type |
PART_NO        VARCHAR2(25)
VNAME          VARCHAR2(25)
ADDRESS        VARCHAR2(45)
PHONE          VARCHAR2(12)
PRICE          FLOAT(126)
PIN            NUMBER(5)

Softwares and interfaces used:

Oracle 7.0
Microsoft Access 97.
Developer 2000 - Forms Rel 2.0.

Excel data Part:

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Part Name</th>
<th>Length</th>
<th>Breadth</th>
<th>Height</th>
<th>Functionality</th>
<th>Category</th>
<th>Material</th>
</tr>
</thead>
<tbody>
<tr>
<td>0020-0983-001</td>
<td>Encoder</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Encoder</td>
<td>Control</td>
<td>Regular</td>
</tr>
<tr>
<td>0798-39-003</td>
<td>Motor Model E728</td>
<td>4.7&quot;</td>
<td>2.27&quot;</td>
<td>2.27&quot;</td>
<td>Motor</td>
<td>Electrical</td>
<td>NA</td>
</tr>
<tr>
<td>1010L</td>
<td>1010 Lite Erector Set</td>
<td>8.33&quot;</td>
<td>1.0 Typ</td>
<td>1.0 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>10SWG</td>
<td>Wires Red, Blue, Black</td>
<td>10&quot;</td>
<td>Regular</td>
<td>Regular</td>
<td>Wiring</td>
<td>Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>1515L-1</td>
<td>1515 Lite Erector Set</td>
<td>43.5&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>1515L-2</td>
<td>1515 Lite Erector Set</td>
<td>26.3&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>1515L-3</td>
<td>1515 Lite Erector Set</td>
<td>22.5&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>1515L-4</td>
<td>1515 Lite Erector Set</td>
<td>17.22&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>1515L-5</td>
<td>1515 Lite Erector Set</td>
<td>10.07&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>1515L-6</td>
<td>1515 Lite Erector Set</td>
<td>9.88&quot;</td>
<td>1.5 Typ</td>
<td>1.5 Typ</td>
<td>Framework</td>
<td>Structure</td>
<td>6105-T5</td>
</tr>
<tr>
<td>16SWG</td>
<td>Wires Red, Blue, Black</td>
<td>16&quot;</td>
<td>Regular</td>
<td>Regular</td>
<td>Wiring</td>
<td>Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>3280</td>
<td>Double Economy T Nut</td>
<td>1.875&quot;</td>
<td>0.443&quot;</td>
<td>0.5</td>
<td>Attachments</td>
<td>Structure</td>
<td>Steel</td>
</tr>
<tr>
<td>3320</td>
<td>15 Series -90 Degree Plate</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Framework</td>
<td>Structure</td>
<td>BHSCS</td>
</tr>
<tr>
<td>4101</td>
<td>10 Series Inside Corner Bracket</td>
<td>2.5&quot;</td>
<td>0.875&quot;</td>
<td>2.5&quot;</td>
<td>Framework</td>
<td>Structure</td>
<td>BHSCS</td>
</tr>
<tr>
<td>Item</td>
<td>Description</td>
<td>Width</td>
<td>Height 1</td>
<td>Height 2</td>
<td>Height 3</td>
<td>Category</td>
<td>Subcategory</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------</td>
<td>-------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>-------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>4302</td>
<td>Inside Corner Bracket</td>
<td>3.0&quot;</td>
<td>1.31&quot;</td>
<td>3.0&quot;</td>
<td></td>
<td>Framework Structure</td>
<td>BHSCS</td>
</tr>
<tr>
<td>4350</td>
<td>15 Series 4 Hole Plate</td>
<td>4.5&quot;</td>
<td>3.0&quot;</td>
<td>1.0&quot;</td>
<td></td>
<td>Framework Structure</td>
<td>Flanged BHSCS</td>
</tr>
<tr>
<td>4351</td>
<td>15 Series 5 Hole Plate</td>
<td>4.5&quot;</td>
<td>0.5&quot;</td>
<td>4.5&quot;</td>
<td></td>
<td>Framework Structure</td>
<td>Flanged BHSCS</td>
</tr>
<tr>
<td>90 Series 65</td>
<td>Castor Wheels</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Wheels</td>
<td>NA</td>
</tr>
<tr>
<td>BATTERY</td>
<td>12 Volt Battery</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Power Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>C++</td>
<td>Turbo C++ Software</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Software Software</td>
<td>Regular</td>
</tr>
<tr>
<td>CB8</td>
<td>Battery Connecting Cables</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Cables Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>CN30</td>
<td>Connectors</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Connections</td>
<td>Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>COMPUTER</td>
<td>Pentium II Computer</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Operating System</td>
<td>Computer</td>
<td>NA</td>
</tr>
<tr>
<td>CPS4</td>
<td>Shaft Couplings</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Drive Connectors</td>
<td>Structure</td>
<td>Regular</td>
</tr>
<tr>
<td>DC48A</td>
<td>Advanced Motion Controls</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Controls</td>
<td>Control</td>
</tr>
<tr>
<td>DMC1030</td>
<td>Galil DMC 1030</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Controls</td>
<td>Control</td>
</tr>
<tr>
<td>DR2PN</td>
<td>Drive Wheels</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Wheels</td>
<td>Regular</td>
</tr>
<tr>
<td>EC10</td>
<td>Battery Insulators</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Insulation</td>
<td>Electrical</td>
<td>NA</td>
</tr>
<tr>
<td>FRF-0302U</td>
<td>3-Channel Radio Remote Control System</td>
<td>5.71&quot;</td>
<td>1.5&quot;</td>
<td>0.73&quot;</td>
<td>Emergency Stop</td>
<td>Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>FSTN84</td>
<td>Fasteners</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Fasteners</td>
<td>Structure</td>
<td>NA</td>
</tr>
<tr>
<td>GEARBOX</td>
<td>Gearbox Ratio 20:1</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Control</td>
<td>Gearbox</td>
<td>Regular</td>
</tr>
<tr>
<td>IMC1100</td>
<td>Galil Breakout Board ICM</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Motion Control</td>
<td>Control</td>
<td>NA</td>
</tr>
<tr>
<td>INVERTER</td>
<td>750 Watt Inverter</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Inverter</td>
<td>Electrical</td>
<td>Regular</td>
</tr>
<tr>
<td>ISCAN</td>
<td>ISCAN Video Tractor</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Vision Image Tracker</td>
<td>Regular</td>
</tr>
<tr>
<td>JSK1</td>
<td>Joystick</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Motion Control</td>
<td>Control</td>
</tr>
<tr>
<td>JVC22</td>
<td>JVC Camera</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Vision Cameras</td>
<td>Regular</td>
</tr>
<tr>
<td>KEELS</td>
<td>Plastic Keels</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Fixture</td>
<td>Cover</td>
<td>Plastic</td>
</tr>
<tr>
<td>MAXIM441CPP</td>
<td>8-Channel Video Multiplexer</td>
<td>Regular</td>
<td>Regular</td>
<td>Regular</td>
<td>Maxim Video Switch</td>
<td>Video Switch</td>
<td>Regular</td>
</tr>
<tr>
<td>Part Number</td>
<td>Vendor Name</td>
<td>Address</td>
<td>Zip Code</td>
<td>Telephone No</td>
<td>Price</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------</td>
<td>----------------------------------</td>
<td>----------</td>
<td>--------------</td>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0020-0983-001</td>
<td>Electrocraft</td>
<td>Queen City Supply, 1859 Section Road, Cincinnati, OH</td>
<td>45237</td>
<td>(513)-351-9500</td>
<td>$225.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0798-39-003</td>
<td>J.N.Fauver Company</td>
<td>11253 Williamson Rd,Cincinnati, OH</td>
<td>45241</td>
<td>(513)-247-9900</td>
<td>$900.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010L</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>45459</td>
<td>(937)-433-8128</td>
<td>$4.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10SWG</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>45015</td>
<td>(513)-868-1766</td>
<td>$15.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1515L-1</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>45459</td>
<td>(937)-433-8128</td>
<td>$20.33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1515L-2</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>45459</td>
<td>(937)-433-8128</td>
<td>$13.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1515L-3</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>45459</td>
<td>(937)-433-8128</td>
<td>$11.51</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1515L-4</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>45459</td>
<td>(937)-433-8128</td>
<td>$9.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Part Number</td>
<td>Manufacturer</td>
<td>Address</td>
<td>Phone Number</td>
<td>City, State</td>
<td>Quantity</td>
<td>Unit Price</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>--------------</td>
<td>---------</td>
<td>--------------</td>
<td>-------------</td>
<td>----------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>1515L-5</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$6.47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1515L-6</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$6.05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16SWG</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>(513)-868-1766</td>
<td>$20.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3280</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3320</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$8.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4101</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$3.90</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4302</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$2.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4350</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$5.30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4351</td>
<td>Volker Control</td>
<td>590 Congress Park Drive, Centreville, OH</td>
<td>(937)-433-8128</td>
<td>$6.75</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90 Series 65</td>
<td>Borne Inc</td>
<td>1526 Scott St, Covington Ky</td>
<td>(601)-291-5447</td>
<td>$28.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BATTERY</td>
<td>Michel Tire</td>
<td>Fairfield, Dixie Hwy-Hamilton,OH</td>
<td>(513)-868-1766</td>
<td>$65.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C++</td>
<td>UC Bookstore</td>
<td>University of Cincinnati OH</td>
<td>(513)-556-0000</td>
<td>$110.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CB8</td>
<td>Michel Tire</td>
<td>Fairfield, Dixie Hwy-Hamilton,OH</td>
<td>(513)-874-8808</td>
<td>$2.14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CN30</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>(513)-868-1766</td>
<td>$0.15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPUTER</td>
<td>UC Bookstore</td>
<td>University of Cincinnati OH</td>
<td>(513)-556-0000</td>
<td>$0.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPS4</td>
<td>Grainger Inc</td>
<td>4420 Glendale, Milford OH</td>
<td>(513)-563-7100</td>
<td>$6.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DC48A</td>
<td>J.N.Fauver Company</td>
<td>11253 Williamson Rd,Cincinnati, OH</td>
<td>(513)-247-9900</td>
<td>$300.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMC1030</td>
<td>Galil Motion Control Inc</td>
<td>575 Maude Control Inc, Sunnyvale -CA</td>
<td>(800)-377-6329</td>
<td>$900.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DR2PN</td>
<td>Borne Inc</td>
<td>1526 Scott St, Covington Ky</td>
<td>(601)-291-5447</td>
<td>$30.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EC10</td>
<td>Michel Tire</td>
<td>Fairfield, Dixie Hwy-Hamilton,OH</td>
<td>(513)-874-8808</td>
<td>$1.50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRF-0302U</td>
<td>Futaba Corp of America</td>
<td>1605 Penny Lane, Schaumburg, IL</td>
<td>(847)-884-1444</td>
<td>$520.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSTN84</td>
<td>OfficeMax</td>
<td>10166 Colerain Av,Oh</td>
<td>(513)-385-5409</td>
<td>$1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEARBOX</td>
<td>Boston gearboxes</td>
<td>7970 Hwy 25, Florence Ky</td>
<td>(601)-525-8021</td>
<td>$340.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMC1100</td>
<td>Galil Motion Control Inc</td>
<td>575 Maude Control Inc, Sunnyvale -CA</td>
<td>(800)-377-6329</td>
<td>$150.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INVERTER</td>
<td>Reliance electric</td>
<td>6950 Washington Av, Minnesota</td>
<td>(612)-942-3600</td>
<td>$600.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor</td>
<td>Contact</td>
<td>Address</td>
<td>Phone</td>
<td>Fax</td>
<td>Total</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
<td>-------------------------------------------</td>
<td>--------</td>
<td>-------</td>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISCAN</td>
<td>ISCAN Inc.</td>
<td>89 Cambridge Street,Burlington, MA</td>
<td>1803</td>
<td>(781)-273-4455</td>
<td>$9,000.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JSK1</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>45015</td>
<td>(513)-868-1766</td>
<td>$30.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JVC22</td>
<td>BestBuy</td>
<td>865 Kemper Rd,Springdale-OH</td>
<td>45245</td>
<td>(513)-671-4305</td>
<td>$400.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KEELS</td>
<td>Cincinnati Plastics</td>
<td>4560 Cornell Rd,Blue Ash Oh</td>
<td>45012</td>
<td>(513)-247-9449</td>
<td>$1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAXIM441CP</td>
<td>Maxim Integrated Products</td>
<td>120 San Gabriel Dr, Sunnyvale, CA</td>
<td>94086</td>
<td>(408)-737-7600</td>
<td>$20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAXIM442</td>
<td>Maxim Integrated Products</td>
<td>120 San Gabriel Dr, Sunnyvale, CA</td>
<td>94086</td>
<td>(408)-737-7600</td>
<td>$20.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MME1</td>
<td>J.N.Fauver Company</td>
<td>11253 Williamson Rd, Cincinnati, OH</td>
<td>45241</td>
<td>(513)-247-9900</td>
<td>$652.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSA-12-80</td>
<td>Galil Motion Control Inc</td>
<td>575 Maude Control Inc, Sunnyvale -CA</td>
<td>94086</td>
<td>(800)-377-6329</td>
<td>$225.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2BSM100</td>
<td>Cincinnati Belting</td>
<td>737 W6 Queensgate OH</td>
<td>45203</td>
<td>(513)-621-9050</td>
<td>$0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPKTS2</td>
<td>Polaroid</td>
<td>5705 Red Arrow Highway,Stevensville,MI</td>
<td>49127</td>
<td>(800)-234-8727</td>
<td>$295.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PS1</td>
<td>AutoZone</td>
<td>3146 Reading Rd, Avondale OH</td>
<td>45219</td>
<td>(513)-961-0700</td>
<td>$90.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PS112</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>45015</td>
<td>(513)-868-1766</td>
<td>$0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PXGLASS</td>
<td>Cincinnati Plastics</td>
<td>4560 Cornell Rd,Blue Ash Oh</td>
<td>45012</td>
<td>(513)-247-9449</td>
<td>$60.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RLYSW1</td>
<td>AutoZone</td>
<td>3146 Reading Rd, Avondale OH</td>
<td>45219</td>
<td>(513)-961-0700</td>
<td>$24.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RS232</td>
<td>J.N.Fauver Company</td>
<td>11253 Williamson Rd, Cincinnati, OH</td>
<td>45241</td>
<td>(513)-247-9900</td>
<td>$100.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSW1</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>45015</td>
<td>(513)-868-1766</td>
<td>$35.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SK4</td>
<td>Grainger Inc</td>
<td>4420 Glendale, Milford OH</td>
<td>45013</td>
<td>(513)-563-7100</td>
<td>$4.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW10</td>
<td>RadioShack</td>
<td>2370 Dixie Hwy, Hamilton-OH</td>
<td>45015</td>
<td>(513)-868-1766</td>
<td>$1.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X9-5304-0310</td>
<td>Shinani Kensi</td>
<td>Marlin P.Jones, PO Box 12685, Lake Park, FL</td>
<td>33403</td>
<td>(800)-652-6733</td>
<td>$20.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>