Accounting Database Design And SQL Implementation

David H. Olsen, (E-mail: dolsen@b202.usu.edu), Utah State University

Abstract

This paper reviews database design in the context of an accounting example and illustrates the power of the Structured Query Language (SQL) with several examples. Many microcomputer database systems have become popular and extinct in the last 15 years which has implications for practicing accountants. Each time a microcomputer database system becomes popular, accountants must learn a proprietary method of questioning or querying the database. In contrast, SQL has remained constant and stable with additions or modifications occurring every few years. Thus, a strategy that accountants should use for coping or keeping up with technology is one that includes gaining some familiarity with SQL. A group of generic business and accounting-related entities are presented in a basic design, which in turn, provide the basis for a set of six relational tables with sample data. These tables provide the basis for ten varied SQL statements that are designed to illustrate the power and usefulness of SQL for accountants.

I. Introduction

It has been reported that the accountant's market share for business and financial information is declining (Elliott, 1994). With the increasing widespread use of the Internet, managers, investors and others now have significantly greater access to sophisticated databases that can be queried directly to obtain timely, custom reports.

Accounting data, particularly financial statement data, are typically highly summarized and are often criticized for being too general. In addition, the format used to present the highly summarized financial data is frequently determined with little consultation with information users. Elliott (1994, p. 75) refers to this problem as forcing users to look at the organization "through a keyhole that permits only a small part of the organization to be viewed.".

Many internal and external financial data users need more data than what the accountant provides and have taken proactive steps to obtain information directly from the organization's database. In addition, financial analysts and other users of corporate financial data do not limit their quest for information to official corporate publications. Marton (1987), for example, describes analysts as pollsters who seek information from every reliable source available to them. Elliott (1994) observes that some analysts go directly to company officials to obtain data.

Changes in the business and information environment, such as those described above, are forcing accountants to take a new look at the profession. The implication is that accountants

Readers with comments or questions are encouraged to contact the authors via e-mail.

must adapt to the changing environment. One way in which accountants can adapt is to move from the role of preparers of information to the role of designers, managers, and auditors of database systems.

This paper describes good database design in the context of an accounting example and illustrates the usage of an industry standard query language. The point is also made that a basic database knowledge is important for practicing accountants. This knowledge should include a familiarity with entity relationship (ER) modeling, table normalization, and query concepts.

The two main approaches to querying a database are query-by-example (QBE) and structured query language (SQL). While QBE approaches have some appeal because of some ease-of-use features, its implementations tend to be dissimilar and thus, force accountants to learn many different implementations of the same theory. For example, many accounting information systems instructors and practicing accountants have learned and/or taught microcomputer database packages like the following: 1) dBASE, 2) Rbase, 3) Paradox, and/or 4) Access. Combine this with the fact that each of these packages may have first been a Microsoft DOS version before being a Microsoft Windows version and instructors and practicing accountants alike are redundantly learning too many different QBE implementations.

In contrast, the essence of SQL has remained stable for 15 years. Though there have been three major standards revisions (SQL89, SQL92, and SQL3), basic SQL is essentially the same query language (Eisenberg and Melton, 1999). The new standards have mainly focused on newer data types (such as objects) and more powerful functions, but SQL statements that were created 15 years ago nearly always function correctly in current database management systems (DBMS). In summary, if accountants become familiar with the essence of SQL, their knowledge will transcend time and will not become obsolete nearly as fast as historical QBE implementations.

This paper continues with a review of the accounting database literature in Section II. Section III is an introduction to a standard, simplified database design that supports basic business functions. The entities that are derived from the basic business design in Section III are the basis for the normalized tables that are used throughout the rest of the paper. In Section IV, a set of SQL examples are presented that are germane to accounting. The examples include a discussion of some of the differences between standard SQL and Microsoft Access 97. An Access 97 discussion is included because it is so widely used in the accounting field. The examples that were selected make use of the most common, basic elements of SQL. While numerous SQL specialty functions are not illustrated, the ones that make up a vast majority of implemented queries are shown. These should be the ones that will most benefit accountants. Section V includes a discussion of the future of databases and SQL in accounting.

II. Review Of Accounting Database Literature

A principal understanding of database concepts has historically been important in the accounting profession. For example, the AAA Committee on Contemporary Approaches to Teaching AIS (AAA, 1986) recommended that instructors include substantial coverage of database topics in their AIS courses. The extent of database coverage recommended by the Committee includes 1) data coding, 2) file/record design, 3) batch/on-line processing, 4) data structures and file organization, 5) database organization, 6) conceptual data modeling, 7) defining database requirements, 8) model databases, 9) extracting data from databases, and 10) maintenance procedures.

Olsen and Calderon (1996) reported on a more contemporary list of database management systems topics and concepts that included a survey to AIS instructors. Those topics are listed in Table 1.

The rationale for proficiency using database management systems (DBMS) is evident from the changing role of accountants as information providers in business and industry. While accountants are still very important providers of financial information to managers and parties external to a business enterprise, their role is rapidly changing (Elliott, 1994). Managers, investors, and others now have significantly greater access to sophisticated databases and can query those databases directly to obtain timely, custom reports. The accountant's role then seems to include more systems analysis and design as well as supporting querying the information system.

Table 1
List of Database Concepts

Alternative database models -- Hierarchical model
Alternative database models -- Network model
Alternative database models -- Relational model
Alternative database models -- Object-Oriented model
Auditing databases
Basic terminology -- attribute (field)
Basic terminology -- entity (record)
Basic terminology -- table (file)
Basic terminology -- transactions
Concurrency control
Data/entity relationships -- one-to-one relationships
Data/entity relationships -- one-to-many relationships
Data/entity relationships -- many-to-many relationships
Data flow diagrams
Data dictionaries
Database management systems software packages
Database administration
Database security
Entity integrity
Entity-Relationship models
Evaluation of databases
General controls for the overall information system
Information systems strategic planning
Information systems user needs and requirements analysis
Network and distributed databases
Normalization -- first normal form
Normalization -- second normal form
Normalization -- third normal form
Normalization -- Boyce Codd normal form
Normalization -- fourth normal form
Normalization -- fifth normal form
Querying a database
Referential integrity
Relational operators
Specific controls for individual applications
Structured Query Language (SQL)
Transaction management

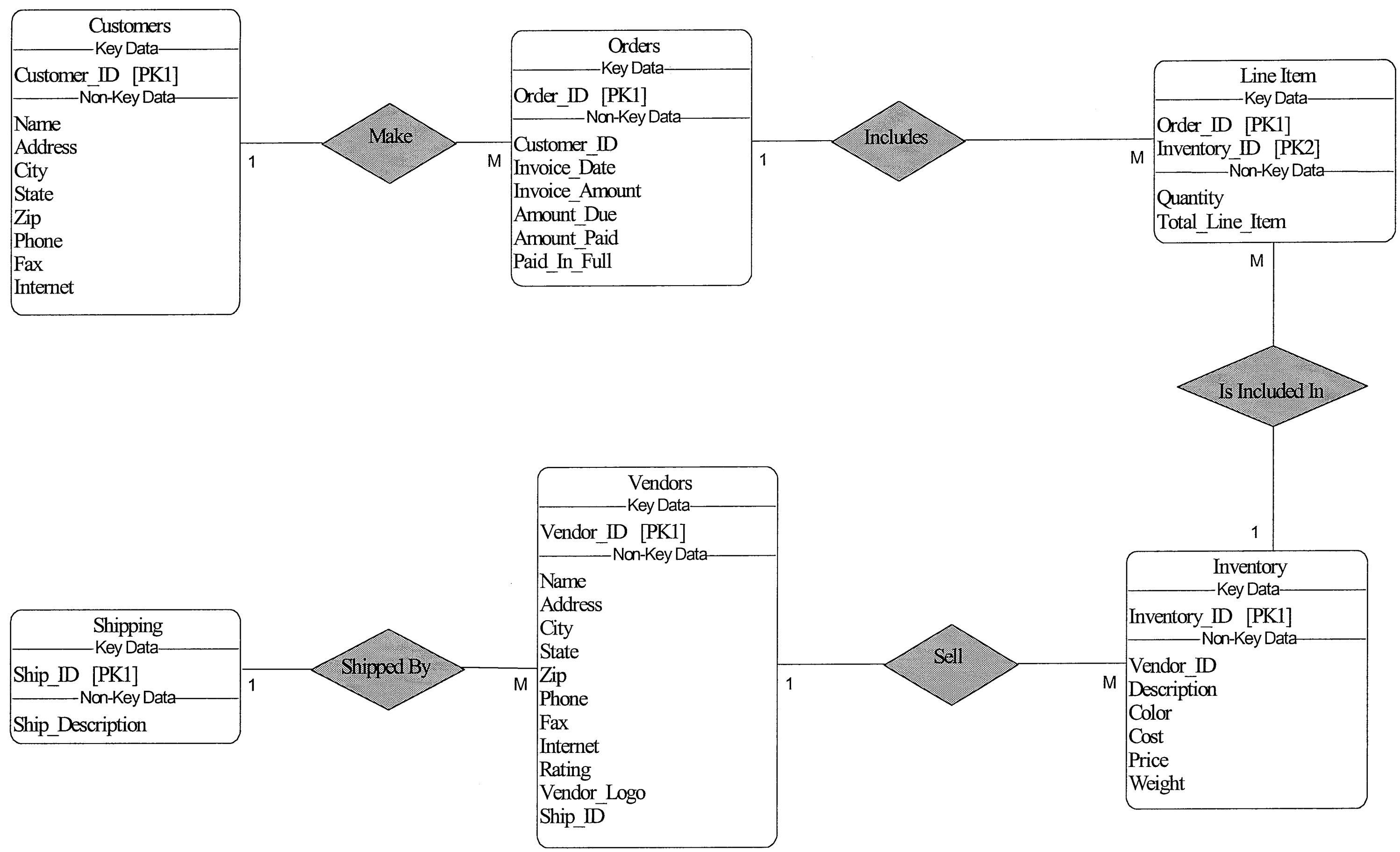
Olsen and Kimmell (1998) reported on advanced database technologies that are either beginning to be used or that may be used in the accounting profession. These technologies include 1) object-oriented databases, 2)Internet databases, 3) data warehousing, 4) data mining, 5) active databases, and 6) business rules. These advanced technologies have important implications for accountants because they will be the basis for future accounting systems. Accountants skilled in these areas will be nicely equipped for the future systems.

III. Database Design And Sql Examples

The following six generic entities are common to most business and are the basis for this presentation: 1. Customers, 2. Orders, 3. Line Items, 4. Inventory, 5. Vendors, and 6. Shipping.

Figure 1 is an Entity Relationship (ER) diagram that conforms to the rules defined by Chen (1976) that illustrates the relationships between the six entities. The Customers entity simply represents customers with the attributes that must be captured. Appropriate customer attributes include the customer's name, address, phone number, etc., and are listed in the table definitions in Appendix A. The Orders entity represents orders that have been placed by customers who already exist in the system. The customers and the orders have a one-to-many (1-M) relationship which means that a customer can place many

Figure 1
Generic Business ER Diagram



orders and that an order is placed by a single customer. Alternatively, the 1-M relationship is sometimes referenced as a parent-child relationship. Figure 1 then includes the number 1 by the Customer entity and an M by the Orders entity.

Similarly, the Orders entity has a 1-M relationship with the Line Items entity. This means that a given order may have several line items, but a single line item belongs to one order. This allows for an Order to contain many different pieces of inventory of varying quantities.

This design also shows a 1-M relationship between the Inventory entity and the Line Items entity, which means that a single type of inventory can appear on many different line items and that a Line Item includes only one type of inventory.

A 1-M relationship also defines the Vendor and the Inventory entities, which means that a vendor can supply numerous different types of inventory and that a type of inventory is supplied by a single vendor. This is admittedly simplistic because most organizations would want to have the option of ordering the same types of inventory from different vendors. Some of the relationships were simplified in order to make these examples manageable.

The final relationship is the one between the Vendors and Shipping entities. We state that a vendor chooses to ship goods with a single carrier such as UPS or Federal Express. To that end, a vendor uses one method of shipping, and a shipper services many vendors. We thus observe a 1-M relationship between the Shipping and Vendors entities.

IV. SQL Examples

The illustrations that are used in the following examples correspond to the six entities that are listed in Section III. Those six entities provide the basis for the six tables that are included in Appendix B. Foreign keys are derived from the relationships between the entities. The rule is "the primary key of the 'one' is placed in the table of the 'many.'" For example, there is a 1-M relationship between the Customers and Orders entities. The primary key of the 1 (Customers) is placed in the table of the M (Orders) as a foreign key. Thus, the orders table has the attribute customer number which provides the link between the customers and orders tables.

Most of the following examples work in Microsoft Access, with the exception of some of the create statements. Some create statements do not work because Microsoft uses proprietary data types. That is, ANSI (America National Standards Institute) standard SQL92 defined a set of standard data types, and Microsoft does not conform to the standard. The ANSI standard SQL92 is used for the following examples but the differences between SQL92 and Access 97 are pointed out.

In Access 97, if a one has already created a query using the standard QBE facility, the user can select the SQL view which displays the SQL version of a program. Otherwise, one can select the SQL view and type in the following queries.

Example 1 - Build the Customers table and put the primary key on the field customer_number field.

Answer:

```
CREATE TABLE customers
(c_num          CHAR(7)        NOT NULL,
c_name          CHAR(30),
c_addr          CHAR(20),
c_city          CHAR(15),
c_state         CHAR(3),
c_zipcode       CHAR(15),
c_phone         CHAR(15),
c_fax           CHAR(15),
c_internet      CHAR(25)
primary_key     (c_num));
```

This example does not work in Access 97 because of the primary key designations. Access uses a proprietary constraint clause to declare a primary key and a foreign key. Access 97 does support the CHAR data type so the rest of the SQL create statement works in Access.

In the second line of the CREATE statement, we declared that null values are not allowed for the c_num field. This means that every customer must have a customer number, and it is not acceptable to leave the field blank. It is common to declare that all primary key fields cannot allow null values. We also declared that the c_num field should be the primary key.

Example 2 – Build the orders table and put the primary key on the o_num field. Make c_num a foreign key that references the c_num field in the customers table.

Answer:

```
CREATE TABLE orders
(o_num          CHAR(6)         NOT NULL,
c_num           CHAR(6),
invoice_date    DATE,
amt_paid        DECIMAL(9,2),
amt_due         DECIMAL(9,2),
paid_in_full    BINARY,
primary_key     (o_num)
foreign_key     (c_num) references customers);
```

This example does not work in Access because of the primary key and foreign key designations. Additionally, Access 97 does not support the DECIMAL datatype, but it does support the NUMBER and CURRENCY datatypes. If CURRENCY were substituted for DECIMAL and the key declarations were not made, this example would create a table in Access.

In this example, we declared that o_num is the primary key of the orders table and we also declared that c_num is a foreign key. This means that a customer number is entered for each order to show which customer made the order. This also makes it possible to join the customers and orders tables at a later time if the user needs information about a given customers order.

Example 3 – List all the information about all the customers

Answer:

```
SELECT c_num, c_name, c_addr, c_city,
c_state, c_zipcode, c_phone, c_fax,c_internet
FROM customers;
```

This query produces a copy of the entire customers table. It is an example of a basic SELECT and FROM query. A basic SELECT clause is used to specify which fields are to be displayed in the answer. Though advanced queries include more complexity in the SELECT clause, this is essentially how it is used. The FROM clause is used to specify which table is being queried. In this case, the SELECT statement was used to choose every field in the customers table and specified that the data was coming from the customers table. The answer table that resulted from this query is identical to the customers table listed in Appendix B.

Example 4 – List all the information about all the customers using the wildcard *.

Answer:

```
SELECT    *
FROM      customers;
```

This gives the same answer as example 3 because it is essentially the same question. The * is used in this example as a wildcard so that all fields in the customers table is selected. Thus, the query essentially states "give me all the records and all the fields in the customers table."

Example 5 - Find all the order numbers from the order table that have been paid in full.

Answer:

```
SELECT o_num
FROM Orders
WHERE paid_in_full = True;
```

This query returns the order numbers of the orders that have been paid in full. This query illustrates using the WHERE clause which is used to "filter" the needed rows or records. In this case, all records that are paid in full are listed, and records that are not paid in full are not listed. The paid_in_full attribute of the Orders table is binary. That is, it can have a value of true or false and is denoted in Access as a checkmark for a true value. The answer that is returned from the Orders tables as listed in Appendix B is:

o_num
0145
1267
3674
4932
5675

Example 6 - Find all the order numbers from the order table that have not yet been paid in full.

Answer:

```
SELECT o_num
FROM Orders
WHERE paid_in_full = False;
```

This query is simply the inverse of the previous one so that all orders that have not yet been paid will be listed. In Access then, this query will return all orders where there is not a check in the checkbox.

Order_Number
4567
5674

Example 7 - Find the order numbers from the orders table where the amount due is less than $10.

Answer:

```
SELECT o_num
FROM Orders
WHERE amt_due < 10;
```

This query returns just the order number attribute for all orders that have the amount due attribute being less than $10. It is important to note that this query will not work unless the amt_due attribute is some type of numerical data type. It could be currency or numeric in Access, but it could not be text. Likewise, it could be decimal in SQL92 but not char or varchar. The basic rule is that if the data will be used in an arithmetical calculation, it must be some type of number. Attributes like social security numbers and part numbers are frequently defined as character or text because these numbers are not used for arithmetical calculations and character or text uses less storage space than number data types.

o_num
3674
5675

Example 8 - Find the vendor name and city from the vendor table where the vendor city starts with the letter L.

Answer:

```
SELECT v_name, v_city
FROM Vendor
WHERE v_city LIKE "L%";
```

This kind of query is very useful when someone remembers some part of a name but cannot exactly recall the name. The " " are used around the letter L because the vendor city is of type text. The % sign follows the letter L indicating that cities have to begin with the letter L and can end with any letter or letters. If one wanted to find all cities with the name "james" in it, we would write LIKE "%james%" which would include Jamestown and St. James.

v_name	v_city
Sanchez's Sup-	Logan
Lisa's Fabrics	Lyman

Example 9 – Find the customer number, name, order invoice amount, amount paid, and amount due where amount due is greater than $0.

Answer:

```
SELECT o.c_num, c_name, i_amt, amt_paid, amt_due
FROM Orders o, Customers c
WHERE o.c_num = c.c_num
AND amt_due > 0;
```

This question requires two new elements to be explained. First, we defined Orders to be assigned the letter o for the sake of convenience. The letter o is then known as an alias for the Orders table and the letter c is an alias for the Customers table. An alias is defined in the FROM clause for convenience in the rest of the query. We use the alias abbreviation in the WHERE clause to join the orders table to the customer table which, brings us to the second point. We need orders and customer information so we "join" the two tables when the customer number in the two tables are equal. This allows us to view information for customers that have orders where there is an amount due.

Example 10 – Find the order numbers where a 'snow blower' is one of the line items that was ordered.

Answer:

```
SELECT o.o_num, i.desc
FROM Orders o, LineItem AS l, Inventory i
WHERE o.o_num = l.o_num
AND l.i_num = i.i_num
AND desc = 'snow blower';
```

This query is substantially more complex than the other queries because it combines several elements to solve one problem. First, we decide to list the order number from the Orders table and the inventory description from the Inventory table in the SELECT clause. We can include any fields that we want to in the SELECT clause. We see that the Orders table, the LineItem table and the Inventory table must be included in our solution so we list them in the FROM clause and include an alias for each table. Thus o will represent Orders, l will represent LineItems, and i will represent Inventory.

The Orders table must be joined to the LineItem table which is the first operation in the WHERE clause. The large, temporary table that

c_num	c_name	i_amt	amt_paid	amt_due
12398	Alex's Music Store	$28.00	$23.00	$5.00
13258	Brian's Brain Surgery	$48.00	$40.00	$8.00
33543	Pacos Bill	$100.00	$55.00	$45.00
75433	Bill's Tacos	$54.00	$34.00	$20.00
68954	Carol's Green House	$90.00	$75.00	$15.00
12345	Fred's Auto Shop	$140.00	$15.00	$125.00
12345	Fred's Auto Shop	$36.00	$20.00	$16.00
12345	Fred's Auto Shop	$30.00	$20.00	$10.00

results from joining these two tables is then joined with the Inventory table. These two joint operations are connected with the AND statement. We can include as many conditions as we want in the WHERE clause as long as we connect the statements with the appropriate AND or OR statements. In this case, we must include a second AND connector to include the 'snow blower' condition.

Perhaps a more visually appealing solution would include parentheses to separate conditions in the WHERE clause. In many cases, parentheses in the WHERE clause would be mandatory to ensure correctness. The above example could be rewritten as follows with parentheses:

```
SELECT o.o_num, i.desc
FROM Orders o, LineItem l, Inventory i
WHERE (o.o_num = l.o_num)
AND (l.i_num = i.i_num)
AND (desc = 'snow blower');
```

o_num	desc
0711	snow blower
1321	snow blower

These few SQL examples were designed to show the power of a robust query language. They are not, however, comprehensive in that literally hundreds of examples could be written to demonstrate all of the language constructs.

V. Conclusion

In recent years, there has been a literal explosion in the use of information technology for business use. The accounting profession has already changed in significant ways in response to advances in technology. Nonetheless, change will continue to occur at a rapid rate. Though database technology is not new, its level of usage is pervasive, and innovative database applications continue to appear. Because of the increased usage of database technology in business, accountants would be well served to have some competence with essential database concepts.

Two important database aptitudes include database design and querying a database. The sample design used in this case illustrates using the ER diagramming method to model a sample business system. The ER diagram is the basis for the normalized tables that are eventually created. The author believes database design is the most crucial skill an aspiring accountant can master in the systems area.

For many years, microcomputer database applications relied on proprietary querying schemes as the primary method of "questioning" the database. In contrast, mainframe database systems primarily use SQL as the primary method of "questioning" the database. The SQL method continues to enjoy the advantage of standardization, which means that accountants can learn one method of querying and still be competent with different databases. The SQL examples in this case were designed to illustrate the key points of querying and are in no way a comprehensive set of statements. We also augmented the discussion with some of the differences between SQL92 and Access 97 SQL because Access 97 is widely used in accounting circles.

Future studies in this area could include more extensive cases that illustrate the depth of SQL and a discussion as well as examples that demonstrate the features of the new SQL3 standard. Other future studies could include an empirical analysis of SQL usage in the accounting profession. 🕮

References

1. American Accounting Association, "Report of the AAA Committee on Contemporary Approaches to Teaching Accounting Information Systems." May 1986.
2. Chen, P.P., "The Entity-Relationship Model: Toward a Unified View of Data."

ACM Transactions on Database Systems (January 1976): 9-36.

3. Eisenberg, A., and J. Melton, "SQL:1999 formerly known as SQL 3." *SIGMOD Record* Vol. 28, No. 1, March 1999, pp. 131-138.
4. Elliott, R. K., "The Future of Audits." *Journal of Accountancy*, (September 1994), pp. 74-82.
5. Marton, A. "The Analyst as a Pollster." *Institutional Investor*, (November 1987), pp. 99-101.
6. Olsen, D., and T. Calderon, "Database Coverage in the Accounting Information Systems Course" *Journal of Accounting and Computers Information Systems* (Spring 1996).
7. Olsen, D., and S. Kimmell, "Towards Integrating Advanced Database Concepts into Accounting" *Review of Accounting Information Systems* (Summer, 1998).

Appendix A
Table: customers

Columns		
Name	Type	Size
c_num	Text	7
c_name	Text	30
c_addr	Text	20
c_city	Text	15
c_state	Text	2
c_zipcode	Text	9
c_phone	Text	11
Input Mask:	\(aaa")	"aaa\-aaaa
c_fax	Text	11
c_internet	Text	25
Relationships		
customers	to	Orders
c_num 1	to	c_num M
Attributes:	One-To-Many	

Appendix B

Customers Table

c_num	c_name	c_addr	c_city	c_state	c_zipcod	c_phone	c_fax	c_internet
12345	Fred's Auto Shop	123 South	Logan	UT	84321-	(435) 745-	(435) 745-	Fredsautosh
12398	Alex's Music Store	921 Rose St.	Logan	UT	84322-	(435) 798-	(435) 798-	Alexmusic.o
13258	Brian's Brain	6574	Chicago	IL	45367-	(756) 874-	(123) 874-	Brian@brain
25684	James' Jury	75 East 1200	St. George	UT	87654-	(435) 546-	(435) 546-	James@gam
33543	Pacos Bill	234 Cactus	Amarillo	TX	79101-	(855) 245-	(855) 245-	Bill@cowbo
34219	Viks Apparel	656 N 9888	Minneapoli	MN	55444-	(852) 741-	(852) 741-	Viks@footb
34265	Bill's Towing	345 Oak St	Smithfield	UT	56789-	(756) 874-	(756) 874-	bill@aol.co
44523	Bun Huggers	334 S Main	Flagstaff	AZ	86038-	(602) 587-	(602) 587-	Buns@hamb
45676	Susan Nail's Salon	4980 Oak	Smithfield	UT	84333-	(435) 765-	(435) 765-	Susan@eart
54234	Counts Blood Bank	12	Appleton	WI	54913-	(988) 556-	(988) 556-	Dracula@bl
56943	Nate's Racquet	567 Elm St.	Baton	LA	45637-	(756) 864-	(756) 864-	nate@stock.
59870	Jon's Haircuts	125 South	Logan	UT	84321-	(435) 792-	(435) 456-	Hair@net.co
68954	Carol's Green	44 S 399 E	Charlotte	NC	28217-	(257) 563-	(257) 563-	Green@thu
75433	Bill's Tacos	9873 D St.	Preston	ID	78566-	(756) 874-	(756) 874-	Tacos@net.
84321	Maui Surfboards	43 Beach Ave	Honolulu	HI	96850-	(456) 123-	(456) 123-	Surfin@poin
85477	Witches Brew	1 Broom Dr	Salem	OR	97306-	(312) 587-	(312) 587-	Black@cat.c
88344	Jacks Used Autos	54 N	Roanoke	VA	23173-	(644) 998-	(644) 998-	Cars@drive.
98563	Buffalo Museum	22 Bill Dr	Sioux Falls	SD	57107-	(778) 235-	(778) 235-	Bill@buffalo

Orders Table

o_num	c_num	o_date	i_amt	amt_paid	amt_due	paid_in_full
0145	12398	8/21/83	$28.00	$23.00	$5.00	No
0298	13258	1/14/99	$48.00	$40.00	$8.00	No
0342	34265	2/10/99	$12.00	$12.00	$0.00	Yes
0711	33543	1/19/99	$100.00	$55.00	$45.00	No
1267	59870	11/ 4/94	$90.00	$90.00	$0.00	Yes
1321	98563	2/28/99	$100.00	$100.00	$0.00	Yes
1598	56943	2/12/99	$50.00	$50.00	$0.00	Yes
1693	88344	3/ 6/99	$25.00	$25.00	$0.00	Yes
2111	84321	4/14/99	$14.00	$14.00	$0.00	Yes
2323	75433	4/29/99	$54.00	$34.00	$20.00	No
2415	68954	5/23/99	$90.00	$75.00	$15.00	No
2948	85477	4/ 2/99	$99.00	$99.00	$0.00	Yes
3674	12345	9/17/97	$140.00	$15.00	$125.00	No
4567	12345	10/28/98	$36.00	$20.00	$16.00	No
4932	45676	11/ 4/98	$8.00	$8.00	$0.00	Yes
5674	12345	10/29/98	$30.00	$20.00	$10.00	No
5675	12398	10/28/98	$110.00	$110.00	$0.00	Yes

Inventory Table

i_num	v_num	desc	color	cost	price	weight_
12345	1789	screwdriver	red	$2.00	$6.00	0.10
22257	5464	sand paper	orange	$0.40	$2.00	0.30
23560	5464	drill	yellow	$56.00	$99.00	2.70
33392	1789	sand	white	$10.00	$25.00	100.00
34567	1789	snow shovel	blue	$10.00	$25.00	4.50
35241	1236	garden spade	silver	$1.00	$4.00	1.10
45000	1236	shovel	gray	$5.00	$18.00	2.00
45068	5464	paint brush	brown	$2.00	$6.00	3.00
45678	1789	snow blower	red	$50.00	$100.00	5.00
48573	1789	ladder	silver	$20.00	$60.00	15.00
56789	5464	garden gloves	green	$1.00	$2.00	0.20
66775	1789	white paint	white	$6.00	$20.00	12.00
67890	5464	light bulbs	white	$1.00	$2.00	0.10
78901	1236	nails	grey	$2.00	$6.00	3.00
85947	5464	brown paint	brown	$6.00	$20.00	12.00
86753	1789	hand saw	silver	$5.00	$14.00	4.00
88574	5464	cypress mulch	brown	$5.00	$15.00	100.00
89012	1236	hammer	grey	$2.50	$7.00	2.00
90210	5464	batteries	black	$2.00	$6.00	1.30
95739	1789	flashlight	red	$1.00	$5.00	0.60
95740	1789	flashlight	blue	$1.00	$5.00	0.60

Shipping Table

s_num	s_desc
1	UPS
2	Federal Express
3	U.S. Post Office

Vendor Table

v_nu	v_name	v_addr	v_city	v_st	v_zipc	v_phone	v_fax	v_internet	v_rating	v_logo	s_num
1236	Jon	22145 Oak	Centervill	UT	87543	(435) 267-	(356) 789-	jbrown@earthl	77		1
1789	Sanche	472 South	Logan	UT	84321	(435) 345-	(435) 765-	Juan@aol.com	98		
3487	Lisa's	345 So. Oak	Lyman	OH	87632	(330) 567-	(330) 578-	Lisa@aol.com	99		
5464	Luigi's	345 East	Millville	UT	84563	(456) 234-	(456) 234-	Lu@aol.com	45		3

LineItem Table

o_num	i_num	quantity	total_line_item
0145	23560	4	$396.00
0145	89012	4	$28.00
0298	45068	8	$48.00
0298	56789	5	$10.00
0298	89012	12	$84.00
0342	12345	2	$12.00
0342	22257	6	$12.00
0711	45678	1	$100.00
0711	48573	1	$60.00
0711	90210	5	$30.00
0711	95739	2	$10.00
1267	35241	5	$20.00
1267	45000	5	$90.00
1321	45678	1	$100.00
1598	33392	2	$50.00
1598	45000	2	$36.00
1693	33392	1	$25.00
1693	35241	2	$8.00
1693	45000	5	$90.00
1693	56789	3	$6.00
1693	88574	5	$75.00
2111	86753	1	$14.00
2323	78901	9	$54.00
2323	89012	2	$14.00
2415	90210	15	$90.00
2415	95739	3	$15.00
2948	23560	1	$99.00
3674	22257	12	$24.00
3674	85947	7	$140.00
4567	45000	2	$36.00
4932	22257	4	$8.00
5674	12345	5	$30.00
5675	12345	1	$6.00
5675	95739	22	$110.00