MIGRATING AND CLEANING DATA USING EXCEL: A CASE STUDY

John Wilton, Master's Candidate and Anne Matheus, PhD
Marist College

Abstract: This is a case study from a senior project in an undergraduate Information Systems program. It involves a real agency that could not afford to have the project completed and requested help from a small liberal arts college in their community. This case study is only one part of a larger project that included the development of a relational database and the creation of a web based application. The problems associated with the migration of the old data into the new system seemed insurmountable, but the challenge was accepted and overcome with excellent results using Excel.

Key Words: Data Cleansing, Extraction, Load, Migration, Transformation, Data Quality

INTRODUCTION

In the summer of 2009, a not for profit agency requested help to redesign their database system. The agency, Daily Out Reach (DOR), opened its doors in 1974. Their goal is to help meet the basic needs of individuals and families by providing them with resources that are difficult for them to obtain. They also act as an advocate through the referral and follow-up process for needy individuals and families and find appropriate resources which will enable people's needs to be met on a long-term basis. They strive to promote community awareness of social problems and generate support for improvement of the system's response to human needs. They operate a food bank, lunch program, health and other support services. DOR's funding is based on the number of individuals served, type of service and through donations.

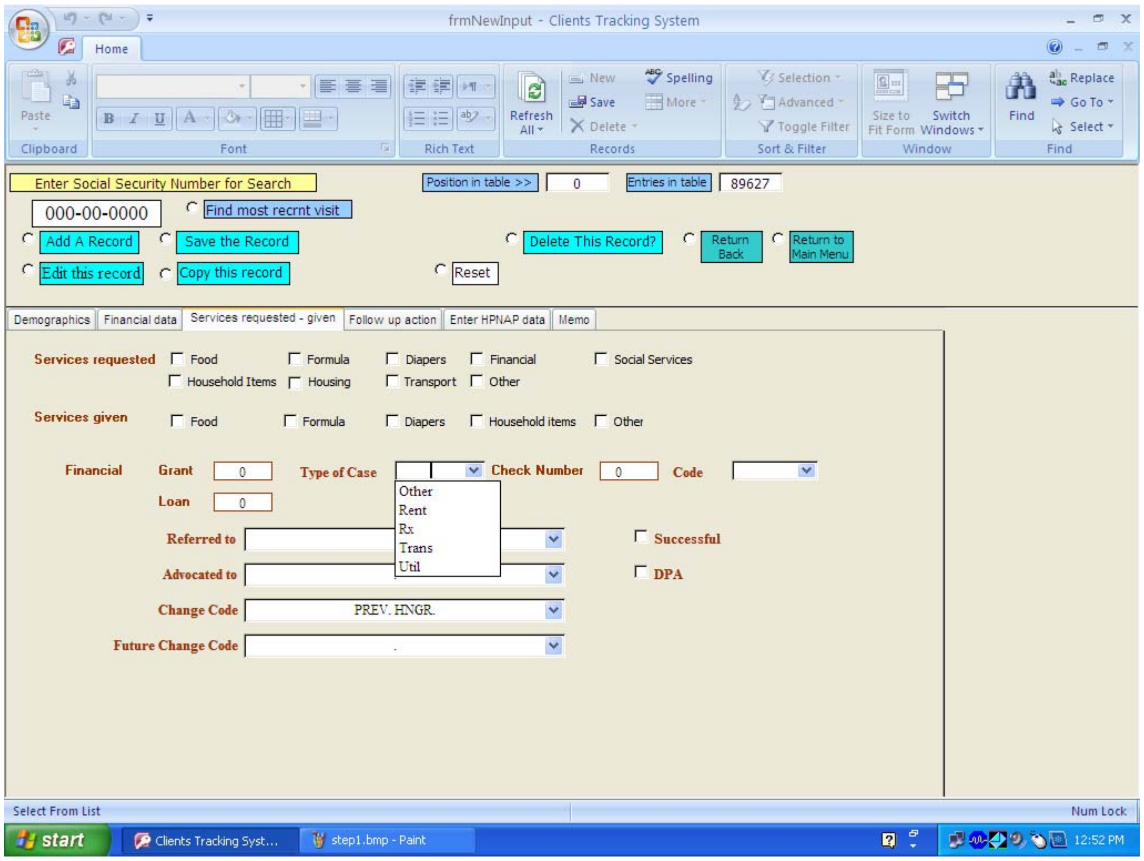
The Problem

The current system to track clients and services is a flat file created in MS Access. It contains over 89,000 records, dating back to 1989. Each time a person visits the agency, a new record is created. It is necessary to enter all the identifying information each time. As a result, much effort is wasted and many of the data fields are duplicated and error prone. It is also almost impossible to determine the relationships between individuals; i.e. to determine if a client is a head of household and to whom he/she is related. Clients are restricted to one monetary grant per household and it is therefore necessary to keep track of the relationships of the people being served.

In addition to requests for funding, this information is necessary to report to sponsoring agencies. Figure 1 is a screen shot of the current system. The following problems will be fixed by creating the new system:

- Inconsistent and redundant data
- Slow access time
- Lack of flexibility
- Need for installation on each PC
- Inability to incorporate future requirements
- Difficulty and potential inaccuracy of tracking and reporting client information

Each record has 110 fields. The structure of the database does not support accurate or timely reporting. Monthly, quarterly and annual reports are frequently tabulated by hand. Funding is dependent upon the accuracy and timeliness of these reports. In addition, third party reimbursement claims are difficult to track and are not generated by the system, but must be manually created, tracked and recorded in hardcopy.



Enter Social Security Number for Search: 000-00-0000

Position in table >> 0

Entries in table 89627

Financial Grant 0

Loan 0

Type of Case: Other, Rent, Rx, Trans, Util

Check Number 0

Code

Referred to

Advocated to

Change Code PREV. HNGR.

Future Change Code

Figure 1: Screenshot of current system

Project Plan

This project has three main phases; the first is to normalize the flat file into a third normal form[1] relational database. This would allow for easy queries, reports and use of the system without requiring redundant data entry and extensive search and retrieval of information. The second phase of the project was to create an easy to use web interface. Finally, the "old" data needed to be migrated into the new system because the history of the services given to clients is required to determine eligibility for future services. A critical aspect of this project was the fact that if the data could not be migrated successfully, the new system could not be used because it was crucial to be able to track the history of the clients' transactions in the system. This case study will focus on the third phase of this project, the data cleansing and migration of the old data into the new system.

However, in order to gain a better understanding of the full project, an Entity Relationship diagram is included as well as screen shots of the web interface. The ER diagram is based on the following business rules:

- One person can have many visits.
- One person can have one or more addresses.
- One person can have multiple sources of income.
- One person can have 0 or more household members.
- One person can have 0 or more requests.
- One person can have multiple requests for different items in one visit.
- One person can have 0 or many service requests.

[1] A two dimensional array containing single–value entries and no duplicate rows; all non key attributes are dependent upon all of the key attributes; and does not have any transitive dependencies. [1]

- One person can have 0 or many stuff requests.
- One person can have 0 or many money requests.
- One person can have different sources of income each time they visit.
- One person can have different contact information each time they visit.

As a result of the normalization of the database, it is now possible to retrieve the following information:

- A list of head of household members.
- The different locations clients have lived in the past.
- Who requested services by dates
  - Broken down further by type of service.
  - Who received services by dates
  - Broken down further by type of service.
- The number of clients served per month or year.
  - Can be grouped and ordered for more structured reports.
- All moneys distributed with reasons, amounts, and check numbers.
- All items distributed with reasons and quantities.
- Demographic information by age group, race, gender, and location.

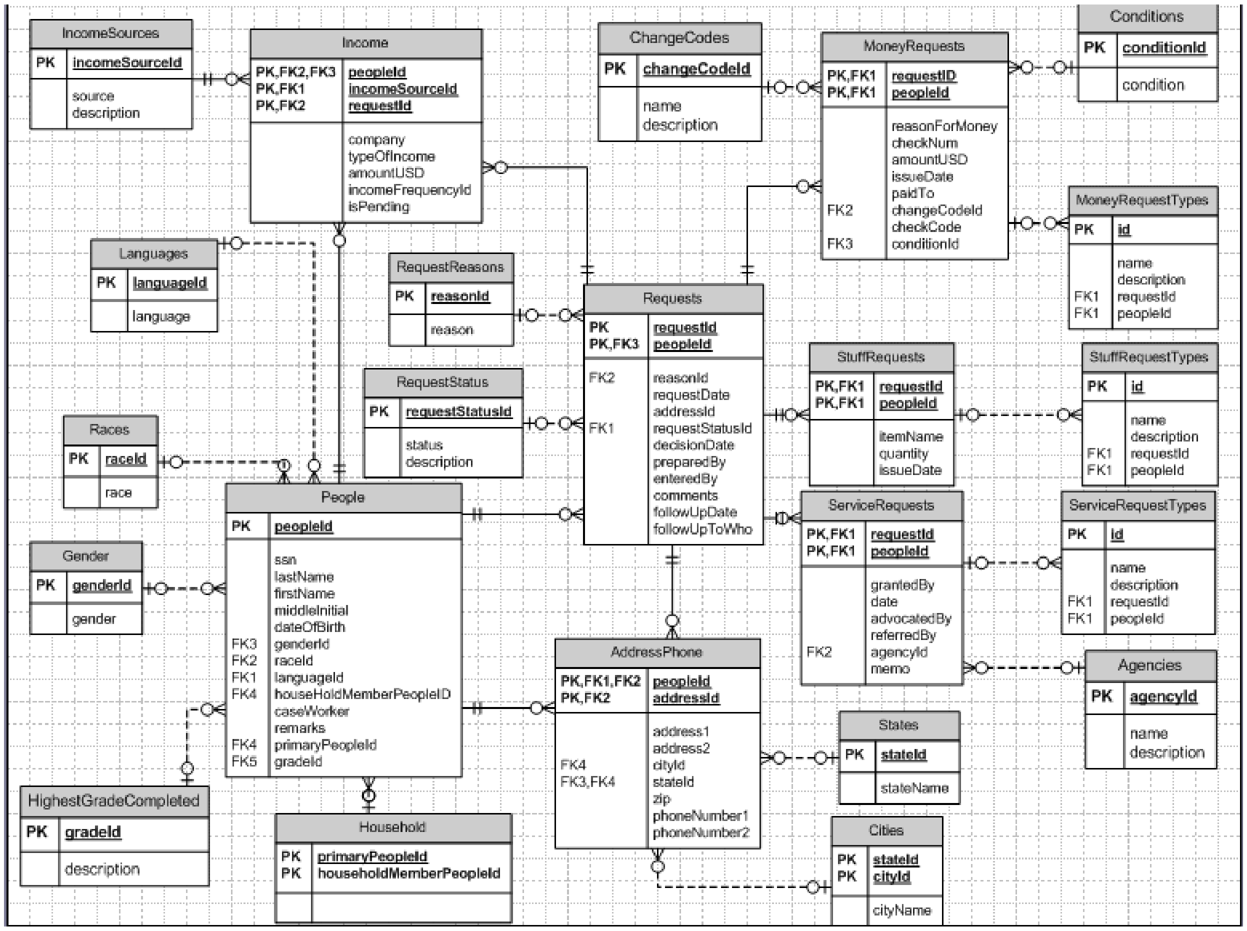


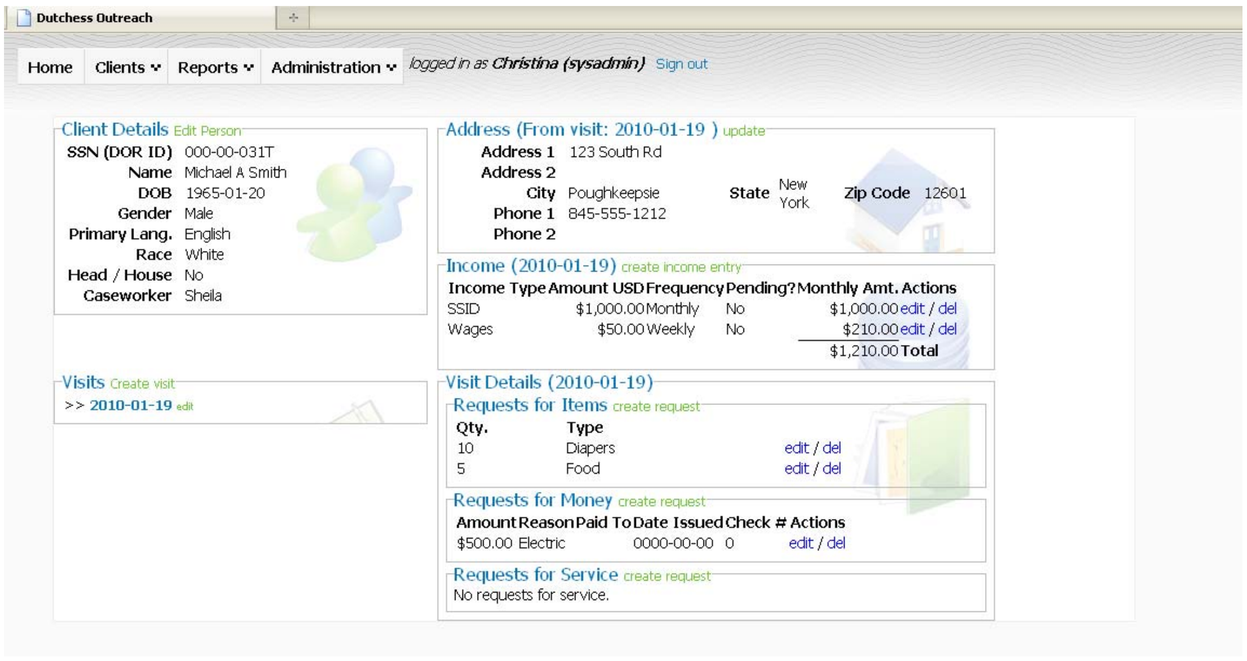
Figure 2: Basic Entity Relationship Diagram

Figure 2 is the basic ER structure without some of the pull down lists for simplicity sake. The final ER diagram includes 33 tables, many of which are simple lists of names and service. The newly designed database is undergoing data migration from the former Access database (flat file) to a relational database.

The original database was based on requests, rather than the people.

User Interface

The new system was created using the Zend framework and coded in PHP. Because of the functionality allowed by the relational database, the system can now display entries for different requests on different dates for easy entry and use. Figure 3 is the main screen with the client and visit information. Below you will notice that the Money request information that was entered is present. Since a date when the check was issued is not entered, it is represented by zeros for now. Once this information is known, the user can go back and edit the request and put in the new information.



Client Details Edit Person

SSN (DOR ID)	000-00-031T
Name	Michael A Smith
DOB	1965-01-20
Gender	Male
Primary Lang.	English
Race	White
Head / House	No
Caseworker	Sheila

Visits Create visit

>> 2010-01-19 edit

Address (From visit: 2010-01-19) update

Address 1	123 South Rd				
Address 2					
City	Poughkeepsie	State	New York	Zip Code	12601
Phone 1	845-555-1212				
Phone 2					

Income (2010-01-19) create income entry

Income Type	Amount USD	Frequency	Pending?	Monthly Amt.	Actions
SSID	$1,000.00	Monthly	No	$1,000.00	edit / del
Wages	$50.00	Weekly	No	$210.00	edit / del
				$1,210.00	Total

Visit Details (2010-01-19)

Requests for Items create request

Qty.	Type	
10	Diapers	edit / del
5	Food	edit / del

Requests for Money create request

Amount	Reason	Paid To	Date Issued	Check #	Actions
$500.00	Electric		0000-00-00	0	edit / del

Requests for Service create request

No requests for service.

Figure 3: User Interface (all data shown is test data and not real clients)

The web-based system was developed to allow for access by simultaneous users. The current system relied on social security numbers to look up clients' names. The new system uses surrogate primary keys[2] to retrieve a client's record.

Data Quality Issues

The original database contains over 89,500 records that include redundancies and errors. In order to validate the data Excel and VBA were used to check records. This involved parsing strings and looking for inconsistencies. Errors within the social security field are an example of some of the errors:

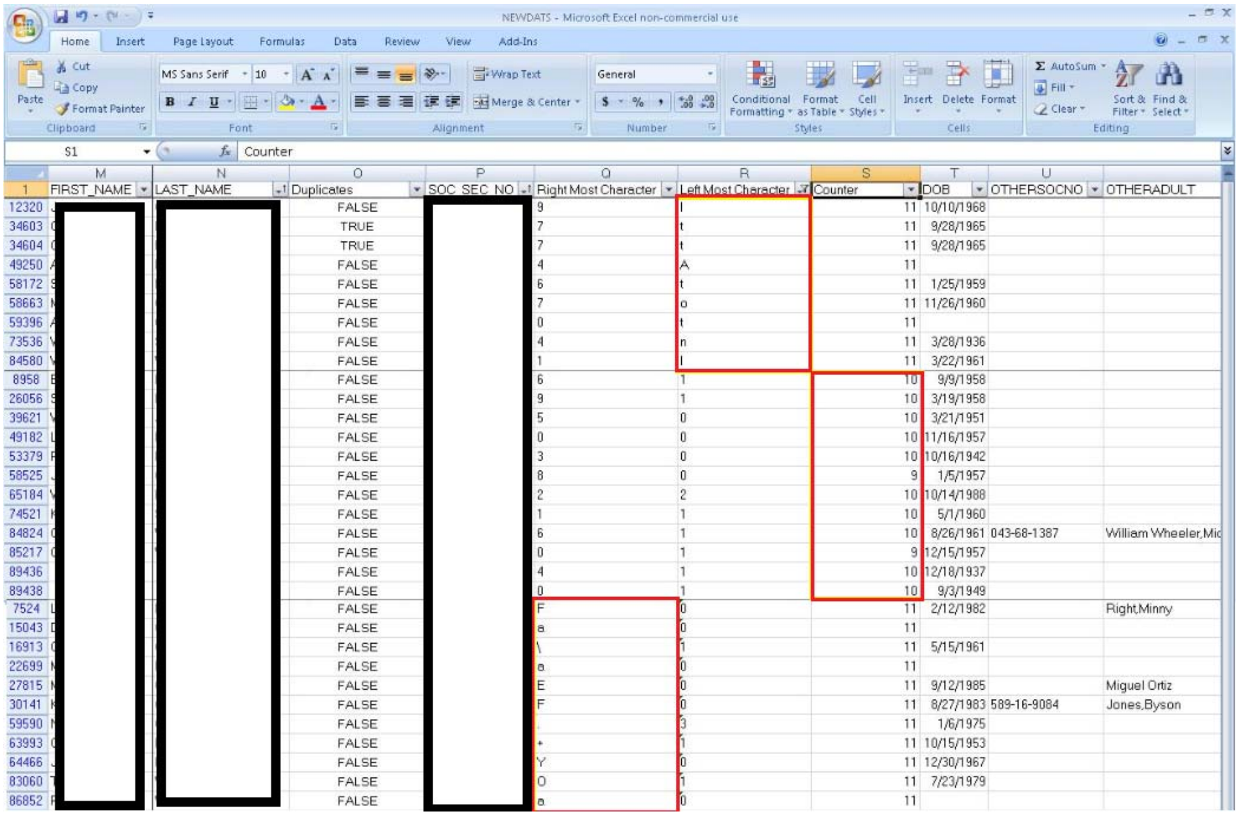
- Check the length of the social security number (SSN)
- In the SSN there would be no characters only digits. This was done by parsing the string and using regular expressions.
- Something that the organization uses is the letter T-Before a made up social security number. (i.e. T12-34-5678) This had to be incorporated in the extraction, transformation and loading (ETL) logic.

In the new system this error is eliminated by adding a checkbox to assure that the users cannot make up a number. If the SSN is unknown, a number is created by auto incrementing and creating a T

[2] A unique, system-supplied identifier that is created for each row and all other attributes are dependent upon it. It has no meaning to the users and is usually hidden from users [1].

number.

Based on the above method of analysis, 79,200 duplicate entries were found; there are 10,300 unique individuals with one person – one visit; these are included in the 20,565 people who had multiple visits.



S1 Counter

1	FIRST_NAME	LAST_NAME	Duplicates	SOC_SEC_NO	Right Most Character	Left Most Character	Counter	DOB	OTHERSOCNO	OTHERADULT
12320	[redacted]	[redacted]	FALSE	[redacted]	9	l	11	10/10/1968		
34603	[redacted]	[redacted]	TRUE	[redacted]	7	t	11	9/28/1965		
34604	[redacted]	[redacted]	TRUE	[redacted]	7	t	11	9/28/1965		
49250	[redacted]	[redacted]	FALSE	[redacted]	4	A	11			
58172	[redacted]	[redacted]	FALSE	[redacted]	6	t	11	1/25/1959		
58663	[redacted]	[redacted]	FALSE	[redacted]	7	o	11	11/26/1960		
59396	[redacted]	[redacted]	FALSE	[redacted]	0	t	11			
73536	[redacted]	[redacted]	FALSE	[redacted]	4	n	11	3/28/1936		
84580	[redacted]	[redacted]	FALSE	[redacted]	1	l	11	3/22/1961		
8958	[redacted]	[redacted]	FALSE	[redacted]	6	1	10	9/9/1958		
26056	[redacted]	[redacted]	FALSE	[redacted]	9	1	10	3/19/1958		
39621	[redacted]	[redacted]	FALSE	[redacted]	5	0	10	3/21/1951		
49182	[redacted]	[redacted]	FALSE	[redacted]	0	0	10	11/16/1957		
53379	[redacted]	[redacted]	FALSE	[redacted]	3	0	10	10/16/1942		
58525	[redacted]	[redacted]	FALSE	[redacted]	8	0	9	1/5/1957		
65184	[redacted]	[redacted]	FALSE	[redacted]	2	2	10	10/14/1988		
74521	[redacted]	[redacted]	FALSE	[redacted]	1	1	10	5/1/1960		
84824	[redacted]	[redacted]	FALSE	[redacted]	6	1	10	8/26/1961	043-68-1387	William Wheeler,Mic
85217	[redacted]	[redacted]	FALSE	[redacted]	0	1	9	12/15/1957		
89436	[redacted]	[redacted]	FALSE	[redacted]	4	1	10	12/18/1937		
89438	[redacted]	[redacted]	FALSE	[redacted]	0	1	10	9/3/1949		
7524	[redacted]	[redacted]	FALSE	[redacted]	F	0	11	2/12/1982		Right,Minny
15043	[redacted]	[redacted]	FALSE	[redacted]	a	0	11			
16913	[redacted]	[redacted]	FALSE	[redacted]	\	1	11	5/15/1961		
22699	[redacted]	[redacted]	FALSE	[redacted]	a	0	11			
27815	[redacted]	[redacted]	FALSE	[redacted]	E	0	11	9/12/1985		Miguel Ortiz
30141	[redacted]	[redacted]	FALSE	[redacted]	F	0	11	8/27/1983	589-16-9084	Jones,Byson
59590	[redacted]	[redacted]	FALSE	[redacted]	.	3	11	1/6/1975		
63993	[redacted]	[redacted]	FALSE	[redacted]	+	1	11	10/15/1953		
64466	[redacted]	[redacted]	FALSE	[redacted]	Y	0	11	12/30/1967		
83060	[redacted]	[redacted]	FALSE	[redacted]	O	1	11	7/23/1979		
86852	[redacted]	[redacted]	FALSE	[redacted]	a	0	11			

Figure 4: Data Errors (names and SSNs have been removed)

In the discipline of information systems, [2, 3, 4, 5], have done extensive work in defining data quality. In their initial work in defining data quality dimensions, [3] used a factor analysis technique to determine the characteristics that users attribute to data. The initial process resulted in 179 dimensions. Using an additional sorting process, the data quality dimensions were grouped into four categories [4]. Category 1 is intrinsic data quality; intrinsic implies that data has quality in its own right. Contextual is the next category, and includes the importance of considering the data quality in relation to the context of the task. Category 3 is representative, which means that the data is concise, consistent, and easy to manipulate. Accessibility is the fourth category and refers to the security and accessibility of the data. The two categories that are the most problematic in this project are the intrinsic category and the accessibility category. The intrinsic category of information quality deals with 4 dimensions. Those 4 dimensions are accuracy, believability, objectivity and reputation. The accessibility category deals with the 2 dimensions. Those 2 dimensions are access and security dimensions. The intrinsic category is best explained "When the quality of the data is directly knowable from the data, then the quality is said to be intrinsic" [6, p 43].

The intrinsic data was not accurate because of end user interaction with the system. The end users had to input clients' information into the system with no form of front end or back end validation. Examples of this are rampant throughout the data, everything from names having numbers in them to letters within social security numbers. Some of the social security numbers were not even 9 digits, this could have been prevented easily through validation techniques.

The believability of the data was always in question because of the inaccuracy of the data. In addition to the inaccuracy there was no accountability for who entered the records. There were only 2 accounts with multiple end users using those accounts. There was an administrator and something similar to a guest account. Unfortunately with no one being accountable it lead to more errors than expected.

The objectivity of the data is compromised. The case worker who is the end user adds an element of subjectivity to everything he or she does. The end user might make an exception for a client that is manipulating the system. Many times clients come in and are going through hard times, and other times they are exploiting this organization. That is something for the end user to ascertain through his or her own experience.

The reputation of the data is one of the worst dimensions throughout the old database. This system has no way of tracking who enters data into the database. Many times a client gives another social security number and claims he is one person; meanwhile the case worker knows that is not accurate information given by the client. Multiple records exist in the system for one client because of fictitious social security numbers. Multiple records for one client lead to degradation of the reputation of the data.

The accessibility category has the access dimension within it. This is a major problem within the older system. One end user can access the system at once. Multiple users can not log in which leads to problems when multiple end users need to access data about the clients. Clients sometimes come in to speak with someone about their hardships and the end user will actually hand write the visit. After the system is available the end user will access the system and enter the data.

The security dimension is nonexistent within this system. If the user knows the location of the executable file on the server they can access the program. Once a drive is mapped, the location is compromised because there is no user login. The security of the system is dependent on end users logging off their computers whenever they step away from their computers.

ETL PROCESS

Extraction

The way the data was extracted was through concatenation of strings. An excel spreadsheet was created that has a custom module. This module creates SQL insert statements that added all the records to the proper fields.

Steps are as follows:

1. Identify what the 110 field headers definitions were
2. Match up the 110 to our new relational database with 33 tables
3. The records would not match exactly. To reduce errors, use as many drop down boxes as possible. Instead of typing male or female, you choose one. Male will reflect a 1 while female will be 0. So while converting the records you must account for this new implementation. This was done in multiple fields.
4. The records must be sorted in order of SSN and then by date. Then the custom module goes through the records and assigns a Primary ID, a Request ID and a Service ID.
5. The above numbers are handled as placeholders /identifiers. These are what determine the records location with regards to the Person object. The Person object is the parent while each Request ID must fall under the Primary ID. This is done by sorting and auto incrementing until the SSN changes.
6. The above determines which SQL statement will be executed and if the SSN is the primary ID or a visit from the absence of a Primary ID and the presence of a Request ID. The SQL statements work off of whatever is generated when the code checks if the below SSN is the same as the above SSN.
7. The Insert statement is generated depending on the SSN. Parameters are matched up with the values from the field headers.
8. To handle replaced text with integers, execute a find and replace macro in each row as needed.

Figure 5 displays the complexity of determining which attribute relates to the original fields, not a trivial process.

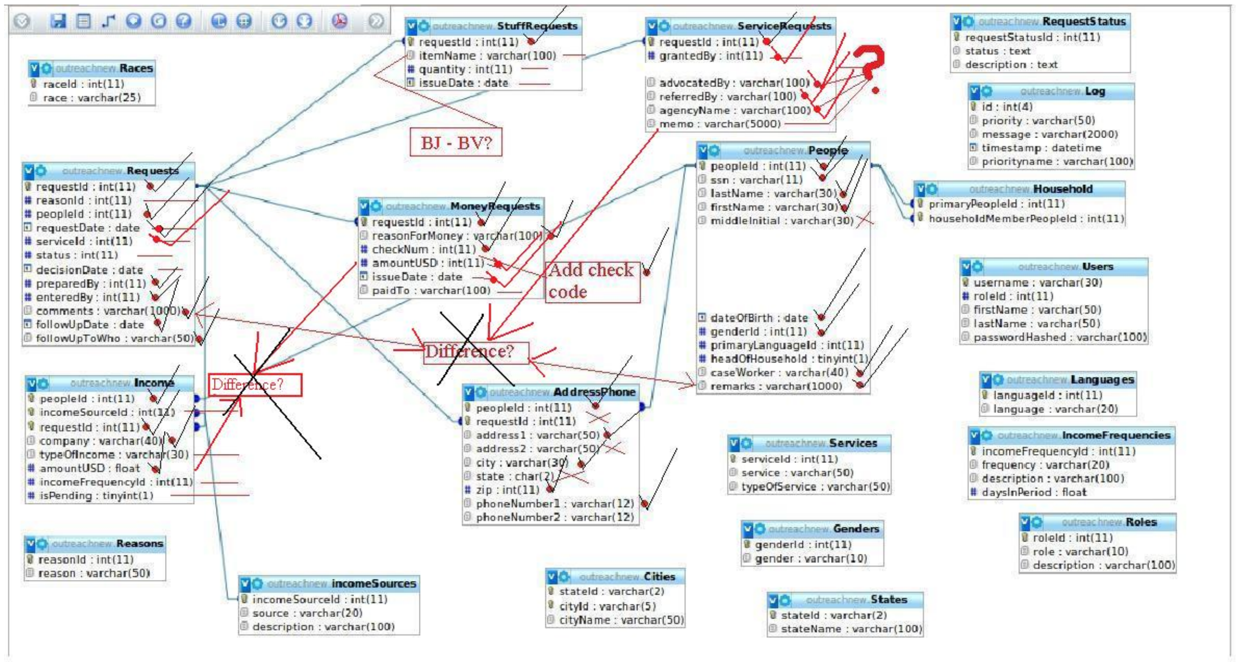


Figure 5: Coordinating attributes between relations.

Transformation

The transfer process was a semi-simple process. We handled this is by taking the above SQL statements and executing them. This created the 33 tables that made up our relational database. The transfer must account for different fields in the database (i.e. the spreadsheet has a String while the database uses a date data type.). These are not handled the same way so it is necessary to use the CStr function to convert dates to Strings for the SQL transfer. This is the same with auto incremented numbers as well as integers and doubles. The SQL only handles Strings so the SQL needs to account for that. This is done by the constraints set in the fields according to the data types.

In addition to having drop down menus to reduce data entry problems there is also JavaScript validation to handle input errors. The records are being converted from a single table into a relational database. This has been done through deconstructing and analyzing the data. This data is then reorganized and placed into their respective fields and interrelated tables.

Load

When loading the records, it is necessary to turn off back end validation. This allows all records to be entered into the database. The order of the load is the most important aspect.

- First you must load the people to create the parent table
- Then you must load the visits
- Next you must load the individual requests
- The SQL statements that get generated in this order

SQL statements are as follows these are strings that you continue to concatenate after VALUES (with the proper cells.

- INSERT INTO People (peopleID, SocialSecurityNum, LastName, FirstName, MiddleInitial, DateOfBirth, Gender, PrimaryLanguage, HeadHouse, CaseWorker, Remarks) VALUES (

- INSERT INTO addressPhone (peopleID, requestID, address1, address2, city, state, zip, phoneNumber1, phoneNumber2) VALUES (
- INSERT INTO Requests (requestID, reasonID, peopleID, requestDate, serviceID, status, decisionDate, preparedBy, enteredBy, comments, followUpDate, followUpToWho) VALUES (
- INSERT INTO IncomeSources (incomeSourceID, Source, Description) VALUES ("
- INSERT INTO MoneyRequests (requestID, reasonForMoney, checkNum, amountUSD, issueDate, paidTo) VALUES (
- INSERT INTO StuffRequests (requestID, itemName, quantity, issueDate) VALUES (
- INSERT INTO ServiceRequests (requestID, date, advocatedBy, referredBy, agencyName, memo) VALUES (
- INSERT INTO Household (primaryPeopleId, householdMemberPeopleId) VALUES (

These values are grabbed from the spreadsheet which has all values from the database.

Lessons Learned

1. Basing the system off of the visit, NOT the individual.

Through creating a new system that is based off of the visit, managing the information is made very clear. For example, if a person makes several visits with a short time frame, the person is only counted once, while the visits will be shown as more instances. This does not mean that the numbers will be skewed or redundant.

2. Use of a relational Database to manage information

Through the use of a relational database, like MySQL, information can be organized and arranged to meet each of the administrative needs. This technology can also prevent unintended access from being granted to those without rights. This was accomplished through the use of a browser based interface which will both vet the information and provide limited Caseworker interaction. Through the use of drop down menus, much of the data will be entered correctly and within the correct category.
Example of this would be someone placing a service request for food and having the person place a service request for diapers. While there will be some human error, a lot of this will be avoided through labeling the fields and having drop down menus that specify what is being done.

3. Use of the server and Online capabilities.

By utilizing MySQL, enabled the user (backend, not the clients) to simultaneously access the system and update records without worrying about interference from one another. This means that many users can access various parts of the system without being locked out.

4. Reporting of the information

Through the use of MySQL to generate reports. The Administration will be able to manipulate and update reports in a very clear and concise fashion. This means that they will no longer need to manually sort through the data and sensitize it so all personal data is removed, that is not required. This also will standardize their reporting techniques and views of information with relative ease.

5. Flat file structures

Creating 33 new tables reduces redundancy and promotes accurate input. This has been a very tedious task and creates new and interesting connections between how the data can be viewed and how it can be manipulated. The organization and interrelations of this program need to be more fully explored to understand how the tables relate to the program as a whole and how to implement (or institute) ways to address future problems and needs.

CONCLUSION

DOR’s current flat file system created an inaccurate and inefficient system that was in need of reorganization. With the introduction of the new system, the granting agencies will have accurate data faster, better information concerning the needs of their clients, how often clients frequent DOR, and can aid in having service requests granted or denied based upon a relational model where the information can be found with a few keystrokes. Speed is an important feature to both the clients and the Administration.

The DOR agency requested a new system created with visual basic with a back end database using SQL Server 2008. The student team felt it would not be as flexible and effective as something written in PHP with a back end in mySQL. After some discussion, the client agreed. Their database was one table with 90,000 records including only 20,000 unique records. Obviously, "redundancy was a major problem". A relational database was created with 33 new tables to reduce redundancy and incorporate input accuracy. Limiting end users' ability to create errors by giving them forced choice lists will significantly reduce data entry errors. This was accomplished by creating a system with drop down menus and validation for the front (JavaScript) and back end (PHP). Table 1 is a synopsis of the identified problems and the implemented solutions.

Table 1: Problems and Solutions

Problems	Solutions
Old system data is a flat file	The old system had no hierarchical structure and had no relationships. The new system is a relational database with 33 tables
Security of old system	There was no login required. Users could find the location of the executable on the server. When records were entered it was under one user name. Now each user must login to the system with a password.
Lack of accuracy of reports Redundancy of records Design of old system	Reports that were generated would count 1 person with 10n visits as 11 people. The people would be entered multiple times due to the visit driven system. The system has now been changed to a people driven system. Each time a person comes to the agency a search is done and if that person is found a visit is added to the parent record if not a new person object is created.
Lack of end user knowledge Inaccuracy of data	Take away end user interaction by replacing data entry fields with drop down lists, check boxes and radio buttons. Example (T-number = made up SSN; Format = Txx-xx-xxx, user inputs ranged from xxx-xx-xxxT to Rxx-xx-xxxx) Reduction of this error is handled by auto incrementing and a check box that generates a T-number.

The following table (Table2) describes the problems in the original database in terms of the actual numbers and rates of errors and problems. As noted, 88% of the records were duplicates; only 12% of the records were unique visits. Less than 25% of the people in the system were not duplicates. The method of finding information in the old system required the end user to literally scroll through all 80,000 plus records to locate a person who may have been in the system in the past. Reports generated by this system could not have been accurate leading to costly and protocol errors.

The new system is designed to eliminate the possibility of duplicate entries by providing the end user with a simple, easily searched database. It is a well-designed relational database that follows the rules of referential and domain integrity. It requires login and has roles assigned to users so that only the administrator role can allow for special functions.

One of the risks of implementing the new system is the data will no longer reflect inflated used of services. Removing the duplicate entries significantly reduces the number of persons receiving services. It was discussed with the administrators at the beginning of the project that this was a possible outcome.

Table 2: A comparison of the old system results and the new system

FACTS & FIGURES		
ORIGINAL DATABASE		
Records: *	89,500	100%
Duplicates:	79,200	88%
Unique: One visit only	10,300	12%
People: Total people in database	20,565	23%
Duplicate People: People who have been to DOR more than once	10,265	11%
Display:	1 table that displayed all relevant information.	
* records (include redundancy and errors)		
NEW DATABASE		
Based on visits rather than people.	Improved storage of client and visit data	
33 tables created from the original database (1 table).	Improved management control	
A better understanding of the data is created.	Improved troubleshooting abilities	
Online Accessible	Provide a visual representation of the system	

REFERENCES

[1] Kroenke, David M. & Auer, David J. (2009). Database Processing. Prentice Hall, NJ.

[2] Wand, Y. and Wang, R.Y. (1996). Anchoring data quality dimensions in ontological foundations. *Communications of the ACM*, 39(11), 86 - 95.

[3] Wang, R. Y. and Strong, D.M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems,* 12(4), 5 - 34.

[4] Strong, D. M., Lee, Y. W. and Wang, R. Y. (1997). Data Quality in Context. *Communications of the ACM*, 40(5), 105 - 110.

[5] Pipino, L. L., Lee, Y. W., and Wang, R.Y. (2002). Data Quality Assessment. *Communications of the ACM*, 45(4), 211 - 218.

[6] Fisher, C., Lauria, E., Chengular-Smith, I. and Wang, R. (2006). Introduction to Information Quality. [s.l.]: M.I.T. Information Quality Program, Print.