P.O. Box 140277
Irving, Texas 75014-0277

Fax: (888) 522-7313
http://www.gentech.org

GENTECH

Genealogical Data Model Phase 1

A Comprehensive Data Model for Genealogical Research and Analysis

May 29, 2000

Title Page

Title:	GENTECH Genealogical Data Model
Data Model Version :	1.1
Document Date:	May 29, 2000

Lexicon Phase 1 Participants

The members of the Phase 1 Lexicon Working Group include the following individuals.

Principal Members:	*Robert Charles Anderson*
	Paul Barkley
	Robert Booth
	Birdie Holsclaw
	Robert Velke
	John Vincent Wylie
Additional Contributors:	*Helen F. M. Leary*
	Beau Sharbrough

Table of Contents

GENTECH Genealogical Data Model

A Comprehensive Data Model for Genealogical Research and Analysis

1.0 PROJECT OVERVIEW

1.1 The Origin and History of the Project

The GENTECH Data Modeling Project is an extension of the work done by GENTECH members on the **Lexicon Project**, an attempt to define genealogical data for the purpose of facilitating data exchange among genealogists. After some work on the Lexicon, the group recognized that it is difficult to define genealogical data out of context because of the various ways people interpret common genealogical terms. The group decided that the effort would be better served by defining genealogical data in the context of a logical data model, which is a systems engineering methodology used to define data in an automated data processing system.

It is important to recognize, however, that the group is simply using this methodology to define genealogical data; the group is not designing software. The Lexicon group was careful to make sure the model has *not* been shaped or influenced by the limitations of current software and hardware. We used data modeling as a means to define genealogical data and the relationships between that data in an effort to bring greater understanding to the genealogical community about data issues. While this does not rule out software developers using the model to create new generations of genealogical software—and in fact the Lexicon group would be delighted if that happens—that was not our goal. As a practical matter, we expect this explicit definition of genealogical data to foster discussion of genealogical data and perhaps in the future to help the genealogical community better exchange data by understanding the limitations of various subsets of genealogical data that may be implemented in automated or manual systems.

APPENDIX A: PRINCIPLES OF LOGICAL DATA MODELING (page 80) contains a discussion of data modeling concepts for readers who may not be familiar with the terminology used in entity relationship diagrams. If you have never worked with data models, you may find it useful to read that section in order to understand both the terminology used in the model, and some of the basic principles that underlie the organizational structure that we used to prevent redundancy, among other things.

It is important to note that we created a *logical* data model, and not a *physical* data model. The logical model describes the relationships of genealogical data, but when that model is actually implemented by a developer, the developer may choose to alter the model using certain methodologically accurate transformations to create the physical data model. Typically, transformations are used to reduce the complexity of the code that must be written, or to increase the performance of the computer system. Clearly, since the purpose of the

Lexicon group is to *define* data and *not* to build an actual system, the logical data model is the appropriate construct.

The group met in Rochester, New York for two days in August 1996 with a facilitator in an intensive working environment called a Joint Application Development (JAD) session, normally intended to bring subject matter experts and developers together. In this case, the developers present were primarily there to act as further subject matter experts, and to facilitate the group's understanding of the data modeling methodology.

During this JAD session, however, it became apparent that although the group was not creating a data model as part of the specifications for a real, to-be-built, specific genealogical application, certain parts of the model could not be created without some understanding of how the data might actually be used in a real application. Thus, the group reluctantly agreed to write a few requirements so that those who study the model can understand the underlying direction. Further, this document attempts to capture some, but not all, of the reasoning behind various portions of the data model. We were hampered by not having a "recorder", the person in JAD sessions responsible for continuously transcribing the results of the discussion. Because this was a volunteer effort and spanned several years, we did not feel that we could afford to have a person in this role; the facilitator brought back the flip charts from each session and transcribed those into this document, subject to group review.

The Lexicon Group met again for two days in January 1997 in Plano, Texas to continue work on the model, and a short meeting was called in May in Valley Forge, Pennsylvania to review our progress. In September 1997 the group met in Dallas, Texas for two days and brought the initial draft data model to closure, although a follow up meeting in Denver in May 1998 was required to complete the data definitions. At that time, it was apparent that the group not only would not finish the data definitions, but had some issues with the current draft as well. A final meeting was held in Silver Spring, Maryland in June 1998. This paper reflects the thinking of the group through that period. It is expected that the data model will be revisited after public comments are received; this document is the formal Request for Comments (RFC).

The statement that best characterized the iterative process that the group went through was finally articulated in Silver Spring: "Now that I look at it, I don't like it." The group continually revisited data model sections to test new ideas against previously agreed upon constructs. The result was frustrating at times as old work was re-opened, but the result was to continually refine the model, making it more general, more powerful, and unfortunately somewhat more abstract than the original concepts. We believe that this is an extremely powerful data model that will accommodate a wide variety of genealogical data, but as a reader of this document, you should compare the model against your own understanding of genealogical data and attempt to find places where data cannot be accommodated by the model.

In order to understand the logical genealogical data model and compare it against your experience, however, it's necessary to not only understand the data modeling terms from systems engineering that are defined in Appendix A (page 80 as previously mentioned), it's also critical to understand the genealogical research process as the members of the group understand it. A process flow diagram and a discussion of this are presented in Section 2.0 THE GENEALOGICAL RESEARCH PROCESS FLOW on page 11.

When all comments have been evaluated, it is the intention of the group to disband and to encourage the formation of a Lexicon II group to use the data model to define genealogical terms.

1.2 Sponsors of the Data Model

Although the Lexicon Project began as an initiative of GENTECH, other national genealogical organizations were instrumental in providing support for this project as the data model evolved. Those organizations, in the order that they were able to join with GENTECH in this project, are the following.

- GENTECH (Charter sponsor)
- Federation of Genealogical Societies (FGS) (Charter sponsor)

- New England Historic Genealogical Society (NEHGS)
- National Genealogical Society (NGS)
- American Society of Genealogists (ASG)
- The Association of Professional Genealogists (APG)
- The Board for Certification of Genealogists (BCG)

These societies are currently sponsoring the genealogical data modeling process and not necessarily the product.

1.3 The Character of the Data Model

1.3.1 Fundamental Principles of the Data Model

The intention of the group was to create a data model that would support the following four principles.

1. The purpose of the genealogical data model is to support the genealogical research process.

2. There is one and only one place to put each piece of data, and there exists a place for every piece of genealogical data.

3. Some researchers will not produce all the data that rigorous pursuit of the process will produce.

4. Actual software systems based on the data model should teach, encourage, remind, and assist users to follow the research process to create high quality genealogical research that can be communicated to others.

1.3.2 Points to Note About the Data Model

The four brief statements in the previous section can be expanded to the following points about the data model.

- The data model expresses our understanding, where possible, of all genealogical data, and it attempts to be completely comprehensive and all inclusive.

- The data model is intended to facilitate the understanding of data issues in the genealogical community, and although the model itself has been created using a systems engineering methodology, the model was not designed to be the platform for a particular piece of software.

- The data model is extensible from our current understanding of genealogical data by putting most kinds of data into tables where rows could be added for additional types of genealogical data that were not considered in the original model. As little as possible is "hard coded" in the model. Thus the model, by being data driven where possible, will accommodate data that we have not considered, but which is of a *type* that we already understand.

- The model should in no way require the genealogical researcher to force data into inappropriate fields simply because the data modelers failed to allow for unusual data. Where the Lexicon group has failed to identify an entire type of data, the model will be extended.

- The data model eventually may support the creation of genealogical software, but the model exists independently of any implementations and does not constrain future developers in their choice of language or hardware, other than to suggest that the relational model is a reasonable construct from which to understand the data.

- The model encourages and supports storing the reasoning behind the genealogical conclusions reached, along with all the evidence that led to those conclusions.

- If a genealogical conclusion is later disproved, the model allows the researcher to correct the conclusion by making a correcting entry, not just purging the originally incorrect conclusion, although it does not force the researcher to correct if they'd rather purge.

- The data model supports both the professional level researcher and the novice by allowing the novice to enter conclusional data without evidence as is currently the widespread practice in genealogical software. Although this is not specifically shown in the data model, the intention is that an actual implementation of this data model would simply fill in place holder records in the intervening entities as needed, in lieu of the novice actually entering the evidence that a more sophisticated user would enter. However, since the model is designed to strongly support evidence, it is anticipated that a sophisticated user interface in an actual software application would strongly encourage the user to enter the evidence, even if entering that data is optional.

- The data model prevents the mixing of other people's data indiscriminately with the researcher's own data. While the model certainly supports the importation of electronic data as it does bringing in more traditional sources, the model also supports the concept of attribution so that no data appears without an audit trail indicating its origin.

- The data model, at a macro level, supports the flow of data from evidence to conclusions through a process of analysis and transformation, and further, supports the continuing use of preliminary conclusions to build more advanced conclusions.

The group deliberately avoided creating lists of legal values for the various entities and attributes, such as surety values, standard repository lists and abbreviations, research objective keywords, and so forth because we felt that trying to set data value standards was beyond the scope of this phase of the Lexicon project.

1.3.3 Following the Data Model

As previously stated, it is our hope that the data model will serve to increase understanding of data issues throughout the genealogical community. Assuming that we have correctly modeled genealogical data, we expect the following.

- **The core of the data model will be followed.** This means that no implementation of the data model will compromise the current entities, attributes, and relationships.

- **The data model will be extended by individuals and companies.** This can be done by the following means.

    - By adding attributes to existing entities.
    - By adding entities to the data model.
    - By adding relationships to the data model, particularly to the new entities.

- **The core of the data model will not be compromised by extensions,** such as removing attributes or entities, or by changing or removing relationships.

1.3.4 Extensions to the Data Model

In addition to the extensions to expert systems discussed in APPENDIX C: DATA MODEL CONNECTIONS FOR EXPERT SYSTEMS on page 94, the group carefully removed a number of entities and relationships after agreeing that they were not needed in the core model. The following are some logical extensions to the model.

- Research objectives can be controlled against a RESEARCH-OBJECTIVE-KEYWORD lookup table.

- Research objectives can be placed in a hierarchical structure so that some high level research objectives have two or more lower level objectives.

- People who appear in citations such as authors, editors, and compilers can be linked to people in the PERSONA entity so that searches that return people who are the subjects of assertions can also return those same people who are involved as authors or in other roles in SOURCEs.

- Both SEARCH and ADMIN –TASK could have attributes for Cost and Time to track expenses and effort. While this would be of interest to professional genealogists who bill for their services, it might be of interest to others as well.

1.4 Frequently Asked Questions (FAQ)

Before we discuss the genealogical research process and the comprehensive genealogical data model that supports the rigorous research process, we address several frequently asked questions (FAQ).

1.4.1 Is this data model a replacement for GEDCOM?

No. GEDCOM is a standard for exchanging data between genealogical applications. Underlying the GEDCOM specification as it evolved over the years has been an implied data model that makes certain assumptions about the relationship of genealogical data. Starting a few years ago, those data models have been made available to the public for inspection. In April 1998, the LDS church released the "GEDCOM (Future Direction) Information Model".

The GENTECH data model is an explicit statement of how all genealogical data is related, and it is keyed to our understanding of the genealogical research and analysis process. If there were a GENTECH data exchange specification based on this model, it could be compared to GEDCOM, but we believe that more understanding is achieved by discussing a logical data model and not just discussing part of a physical implementation of a model.

Thus, this data model cannot be compared to GEDCOM because they are two different constructs. The GENTECH data model, however, can be compared to the "GEDCOM (Future Direction) Information Model" although the methodology used to actually draw each model is considerably different. It should be noted that genealogical data in toto is sufficiently complex that some interpretation of the data relationships is to be expected, and thus two teams are not likely to draw the model exactly the same way, even if both teams have an identical understanding of the data. Both models share many features, but differ in other ways.

1.4.2 Where are the GEDCOM tags?

Genealogists who use computers to store their genealogical data are often familiar with the tags used in GEDCOM. As explained in the previous section, GEDCOM and the GENTECH data model cannot be directly compared because they are different constructs, but a logical question is, "Where is the information stored in this data model that would otherwise be found in GEDCOM tags?"

That information exists in a number of attributes throughout the model, and in fact will be the focus of the second Lexicon group. The following are the attributes that contain "tag-like" data.

- Citation-Part-Type-Name
- Representation-Type-Name
- Place-Part-Type-Name
- Event-Type-Name
- Event-Type-Role-Name
- Group-Type-Name
- Group-Type-Role-Name
- Characteristic-Part-Name
- Characteristic-Part-Type-Name

1.4.3 Where is the citation, and where is the family?

It became apparent during our sessions that it is very easy for people to confuse a *data view* with a *specific entity*. The most easily seen example of this is the concept of a **citation**. Part of a typical citation involves information about an entire source of records, such as a will book in a county court house, or a book of depositions. The researcher makes note of certain information such as the title of the source, and where it was found, that is, what repository it was found in, and what the call number was.

Clearly, a single book of wills, for example, will contain potentially many wills of interest to a researcher. For a given will, there will be an additional part of the citation that refers to the page or other reference number that locates that particular record or document in the source. If we use three wills in writing a journal article for example, part of the entire citation—the part that deals with the will book itself—will be redundant information in our citations. The redundancy is to be expected and will be dealt with using various output conventions of the particular journal such as *ibid.* Citing the physical repository, if necessary, also causes us to repeat information.

It is important to recognize that in data modeling, redundancy is *not* a desirable condition (see Appendix A if this does not seem self-evident) and we cannot make it go away by invoking Latin phrases. This paper explains further how we recognize redundancy in data modeling and what we do about it, but for now it's sufficient to understand that we store the repeating part of the citation in one place, where we store information about sources, and we store the non-repeating record portion of the information in another place. In our actual model we have this type of citation information in *three* places: REPOSITORY, SOURCE, and CITATION.

The fact that this citation information does not appear in the model in *one place* should not be cause for alarm. This is the way we express the relationship of data in the formal data modeling methodology, and in fact if the model is implemented in an actual system, it would be the computer's job to pull the pieces together as needed. If the two parts (or three parts) of the citation, in this example, can be seen in entities that are linked together, then we can rely on the developer to write the instructions that tell the computer that a complete citation requires information from more than one place.

This same concept can be extended on through the chain of connected entities, so that information that appears on one side of the model can be combined with information that appears on the other side of the model, *if* the two entities are joined with appropriate relationship connectors. Determining whether a chain exists requires the ability to understand and analyze the relationship connectors, and is a mechanical exercise that an experienced data modeler applies to a new model.

Again, it is incorrect to expect all the data that a researcher would want to see on one screen or in one paragraph of output to appear in a single place in the model. Thus, there is no entity defined for **family** because a family can be constructed by a computer program from the data about the individuals that make up the family. This *family view* can be presented as a report such as a Family Group Sheet, or it can be presented as a display on the screen that lets the user drag names (or icons) representing people into appropriate family groupings. The fact that family is a useful view of data does not mean that the data is actually stored in a single entity called FAMILY. This same rationale can be extended to Birth, Death, Marriage, Burial, and others; while we do not have separate entities for these kinds of data, the model accommodates the efficient storage of this data and it can be retrieved from the model as needed.

APPENDIX B: LOGICAL VIEWS OF THE DATA MODEL on page 93 contains a discussion about some common views, and discusses where the data is found for those views.

1.4.4 Why is the model so complicated?

Another concern of the group is that the data model appears too complicated. It is certainly true that for someone with no technical background in data modeling, understanding the model will be a challenge. That's why we've included some background material in Appendix A to help genealogists who do not have systems engineering credentials.

Beyond the hurdle of understanding the symbols used, it is not always easy to begin thinking about genealogical data at an abstract level. We've included sample instance data to help bridge that gap; many times it was necessary for the group to look at actual data to prove or disprove that we were on the right track.

In addition, the data model is complicated because genealogical data not only spans an enormous variety of record types, it also involves data that must always be viewed with skepticism. A typical business application may have a lot of different kinds of data types, but seldom is the business data model designed to purposefully accept data that later will be proved to lead to incorrect conclusions.

In fact, the idea of modeling data in such a way that additional conclusions can be reached from the stored evidence is itself another complex idea. Storing the chain of reasoning is not a typical business application, and while business decision support systems which deal with conclusions are available, they are not usually mixed with the sort of suspicious data that is normal in genealogy.

Further, some genealogical data, such as relationship data, does not fit into the normal relational model that is the basis of most data modeling. Ancestor and descendant data is clearly tree-like in structure, and doesn't fit the relational model particularly well.

Other peculiarities of genealogical data, such as the unreliability of names as a permanent key to a person's identity, lead us to unusual constructs such as the **persona** (explained in Section 5.4.6 Persona Entities on page 40) which is then manifested in the model and which then requires some study to understand.

Finally, if modeling genealogical data in a comprehensive way were easy, it would already have been done repeatedly in application after application. Since that is clearly not the case, it strongly suggests that the genealogical data model is not intuitively obvious, but has sufficient peculiarities to require substantial study. We hope this RFC will facilitate that study and discussion.

1.4.5 Could such a tedious model ever be used to build a real application?

It is somewhat bothersome to most people to work through the examples of evidence and conclusions, and see how the verbatim extracts are turned into many simple statements and how those statements then become scattered over multiple tables with only terse numeric keys to connect them. A normal reaction is to be concerned that such a data model will be extremely tedious to use, and that after a few hours of entering data, the researcher will never want to use the application again.

It is important to stress, once again, that the Lexicon group has built a data model to facilitate the understanding of genealogical data, not to support any particular software implementation. That said, however, the group was forced to consider whether it was even reasonable to think that this model could eventually become a software application.

Because the data model was intended to support everyone from the beginner to the professional level researcher, it is also easy to believe that it won't meet the needs of *anyone*. It appears to be too complex for the beginner, and perhaps too tedious for the experienced researcher who can easily extract the pertinent information from a document, ignoring the boilerplate, and leap directly to a final conclusion.

It is also important to stress that the way the data model is actually implemented will determine the ease of use. Although the data model bears the same relationship to a computer program as the foundation bears to a house, a good, a solid data model is not sufficient to ensure an easy to use program any more than a solid foundation is sufficient to ensure that a house will be pleasant to live in. Clearly, a bad foundation means that a house can never be built correctly; walls will always be out of alignment and will perhaps shift over time. Similarly, a bad data model guarantees that a program will not meet the needs of the users, and further, it will make it extremely difficult for the programmers to build the application, because they will be building around inherent problems.

Continuing the analogy, for a house to be pleasant the rest of the design must also be executed properly. The stairs should not be too steep or too narrow, the walls and roof should have adequate insulation for the climate, and the colors used for the siding, the walls and ceilings, and the furnishings should be chosen carefully for the needs and tastes of the people who live there. None of these things have anything to do with the foundation but they, too, can ensure an unpleasant environment if not executed correctly.

We believe that there is a sufficient level of expertise in building user interfaces, and that there are software development tools to supply those interfaces, so that the data model can be implemented in such a way as to encourage the researcher to enter the evidence and not just the conclusions. Further, we think a user interface that implements the data model can minimize mistakes and maximize productivity.

One example is the user interface for extracting information from the verbatim transcript. If the researcher chooses to capture the verbatim text of a document, and now wants to extract pieces of that text, the worst way to do this would be for the researcher to type those pieces again. This not only brings into question the wisdom of having typed the entire transcription in the first place, it also greatly increases the chance of making an error. A better user interface would be for the application to let the user highlight the portion of the text from which to make the extract, and let the computer copy that text as needed, much as a word processor lets a user copy selected text. A refinement of the concept would be for the user interface to not actually copy the data again, but simply store the extract as a set of pointers to the original text so that if an error in the transcription is corrected later, any extracts from that section of the transcription are automatically updated as well. The user would not see the pointers, of course, but would see a list of extracts from the transcription, and by perhaps selecting one from the list, would then see the extract highlighted in yellow in place on the screen. This is just one example of how a clever user interface in a real computer program can make this data model highly usable.

1.4.6 What limitations does the model have because of computer technology?

None. The model doesn't have anything to do with computer technology except to borrow a useful defining methodology from systems development work. This is a logical data model and is intended to foster the understanding of the types of, and relationships between, all genealogical data. Although it may serve as the foundation for genealogical software in the future, no technology limitations shaped the model.

1.4.7 How are primary and secondary sources handled in this data model?

Although the issue of primary versus secondary sources is certainly important in understanding the quality and likely reliability of the data being evaluated, we did not find this to be an issue relevant to the data model. While the concept of reliability is crucial, we did not believe it was relevant to try to tag entire data sources as "primary" or "secondary", because clearly the small pieces of data retrieved from those sources may be in some cases primary, and in other cases secondary. A good example of this is a cemetery record. The part of the record dealing with the date and place of death and burial is certainly primary; it was a contemporaneous record made by people who were likely to know the facts. But the part of the record dealing with the birth of the deceased is much more speculative.

In the data model, this issue of reliability is handled with the surety scheme. The researcher determines the appropriate level of surety and works with that rather than an arbitrary standard that may not fit the data. Surety in our model is an attribute of assertions and not sources since data in a source may be of different surety levels when used for different purposes; many sources also contain a mixture of data types of varying degrees of surety.

1.4.8 Will GENTECH certify software that complies with this data model?

No.

It's not the goal of GENTECH and the other data model sponsors to set up a program to certify genealogical applications as being compliant with a particular version of the data model. The work on the data model was undertaken to support the entire genealogical community, and no certification program is anticipated or planned.

If conditions change in the future and it appears beneficial to the genealogical community that such a certification program be put in place, GENTECH and the other sponsors may reconsider the issue.

2.0 THE GENEALOGICAL RESEARCH PROCESS FLOW

Before a data modeling attempt can be successful, it is necessary for both the creators of the model and the readers of the model to understand and agree to the same general process for gathering and analyzing genealogical data. One way to represent this process is to show it as a process flow diagram. There are almost no special engineering symbols to understand, and the process flow diagram will help set the stage for the detailed discussions that follow.

In the diagram inserted after this page, the boxes in the upper portion of the diagram represent steps that the genealogist undertakes in genealogical research, starting with "Strategize for Further Research". Steps that follow from another step are marked with an arrow; some steps, such as "Produce Output Reports and Exports" can happen at any time.

The lower portion of the diagram contains additional information about each of the steps; we say they break down the steps into smaller work pieces.

It is important to notice that when a document has been located, the genealogist analyzes the content, updates the project status in some sort of log, records the citation, and then gathers the data as an image (photocopy or other), by transcribing the text, or by extraction or abstraction. This raw evidence is then used to make simple evidential statements, such as "John Smith was born before 1762." A thorough research process requires the genealogist to take these small, simple evidential statements and combine them into meaningful, well supported conclusions that resolve data discrepancy issues. While more experienced researchers may handle this in a variety of ways, and while novices may fail to capture citation data and one day may have to repeat the research, the data model must (and we think does) accommodate this comprehensive research process.

It is this process of transforming small pieces of evidence into small conclusional statements that is the heart of this data model, and perhaps the most significant contribution of the model to understanding genealogical data. This issue is discussed in more detail Section 3.0 THE UNIFIED THEORY OF GENEALOGICAL DATA on page 12.

3.0 THE UNIFIED THEORY OF GENEALOGICAL DATA

It is important to understand the underlying similarity of all genealogical data. Although genealogists deal with an enormous variety of data sources, one of the central concepts of this data model is that *all* genealogical data can be broken down into a series of short, formal genealogical statements. These statements are shown below using the following abbreviations.

P1 stands for Person Number 1. P2 stands for Person Number 2. In practice, both Person 1 and Person 2 are actually links to a record rather than a specific person name. The PERSONA entity, as it will be demonstrated in the data model section later, contains a link to the actual name.[1]

The term "place" refers to either an explicit place name, or if a place authority entity exists, place means a link to that entity.[2]

The "date range" allows non-specific dates to be used along with specific dates. This means soft dates like "about" or "circa" as well as date ranges.[3]

"Events" are anything that may have happened in someone's life such as birth, death, christening, marriage and so forth.

The term "relationship" allows the capture of explicit relationship information between two people, so that we can say, for example, P1 is the father of P2.

A "characteristic" describes a person, and can be a physical characteristic, a personality trait, or more diffuse data such as occupation. A characteristic is generally a descriptive fact that applies to the person in question over a reasonably long period of time.

There should not be *any* genealogical data that cannot be put into one of these terse statement types, although certain data, such as land boundary descriptions, will be much easier for humans to understand when *presented* in a different format, such as a plat.

It is also important to note that after the group worked with three basic statement types, and in fact continued to create additional statement types, eventually a Super Statement Type evolved. However, since the data model became progressively more abstract as the statement types became more powerful, we chose to discuss the initial three types, and then show the final form without showing all the intermediate steps. For readers interested in the intermediate statement types, however, see APPENDIX D: ADDITIONAL STATEMENT TYPES on page 99.

[1] See Section C.1 PERSON NAME EXPERT SYSTEM on page 94 for a discussion of the complexity of names used in genealogy.
[2] Also see Section C.2 PLACE NAME EXPERT SYSTEM on page 95 for a discussion of the complexity of place names.
[3] Also see Section C.3 DATE EXPERT SYSTEM on page 96 for a discussion of the various kinds of dates used in genealogy.

3.1 The Original Statement Types

There are three statement types that form the basis for the group's original attempt to define all genealogical data.

3.1.1 Statement Type 1: Statements About Relationships

The Relationship Statement has the following form.

P1 (link) / Relationship / P2 (link) / Date Range / Place

An example of the Relationship Statement is the following.

1234 (i.e., George Smith) / is the father of / 2143 (i.e., John Smith) / on October 1, 1847 / in Prince George's Co, MD, USA

3.1.2 Statement Type 2: Statements About Events

The Event Statement has the following form.

P1 (link) / Event / Date Range / Place

An example of the Event Statement is the following.

1234 (i.e., George Smith) / married / July 1, 1845 / in Prince George's Co, MD, USA

3.1.3 Statement Type 3: Statements About Characteristics

The Characteristic Statement has the following form.

P1 (link) / Characteristic / Date Range / Place

An example of the Characteristic Statement is the following.

1234 (i.e., George Smith) / was a blacksmith / in March, 1853 / in Prince George's Co, MD, USA

3.2 Statement Types 1 to 3 Compared

The table below shows how data from all three basic statement types could be combined into a single table, so that the repetitive nature of date and place can be seen.

	P1	*Relation-ship*	*P2*	*Date Range*	*Place*	*Event*	*Char-acteristic*
S1 Statement **Relationship**	John Smith	Is the father of	Robert Smith	07 Jan 1862	Mont-gomery CO, MD, USA		
S2 Statement **Event**	John Smith			22 Jun 1840	Rowan CO, NC, USA	Was born	
S3 Statement **Characteristic**	John Smith			15 Aug 1871	Fannon CO, TX, USA		Was a stage coach driver

The table below shows another way to consider events and characteristics, as types that are matched with a *value*. Thus, in many cases a birth is a birth, but occasionally we might find an indication that the birth was a Caesarian. Occupation is meaningless without a value such as piano tuner, teamster, trapper, or farmer. The range of values of sex is somewhat constrained, but the range of values of age is larger. The concept of values will prove useful later when we discuss the actual data model.

Type	*Date Range*	*Place*	*Value*
Birth	31 Dec 1899	Rowan CO, NC, USA	Caesarian
Occupation	31 Dec 1899	Rowan CO, NC, USA	Piano tuner
Sex	31 Dec 1899	Rowan CO, NC, USA	Male
Census	01 Jun 1870	Rowan CO, NC, USA	U.S. Federal
Age	31 Dec 1899	Rowan CO, NC, USA	42

3.3 The Super Statement Type

The group took the three statement types, and added two additional statement types that are discussed in APPENDIX D: ADDITIONAL STATEMENT TYPES on page 99. The result is a Super Statement type that ties together two of the following: PERSONA, EVENT, GROUP, or CHARACTERISTIC. The following chart, discussed again later in section 5.4 CONCLUSIONAL SUBMODEL on page 29, shows the most likely pairings.

		Subject 2			
		PERSONA	EVENT	GROUP	CHARAC-TERISTIC
Subject 1	PERSONA		✓	✓	✓
	EVENT		✓	✓	
	GROUP	✓		✓	
	CHARAC-TERISTIC			✓	

The Super Statement type uses two "subjects" and a value as follows. Note that we omit date and place for clarity. The chart on the following page identifies a statement form type using our original list of statement types S1, S2, etc. For each of these basic statement types, we have a subject 1 type and a pointer to that particular subject 1. We also have a subject 2 type and a similar pointer to that particular subject 2. Clearly, the first column is redundant information with the two subject types, since a statement form such as S1 is "known" to have two PERSONA subjects. Also note that we use pointers in the table to data elsewhere. The Value column is always a pointer to a table of values for characteristics of occupation, characteristics of hair color, relationships of people, etc.

Note also that the final form taken by the data model is somewhat different from the examples shown on the next page; this has been done to assist the reader in understanding how different kinds of low level statements can be combined. The actual combination in the model, for example, drops the S1 type in favor of a stronger—but somewhat harder to understand—construct.

Subject Form Type	***Subject 1 Type***	***Subject 1 ID***	***Subject 2 Type***	***Subject 2 ID***	***Value (Always a pointer)***	***Notes***
S1	Persona	(Pointer)	Persona	(Pointer)	Father	Subject 1 to Subject 2 relationship.
S2	Persona	(Pointer)	Event	(Pointer to, say, a wedding)	Groom	
S3a	Persona	(Pointer)	Character-istic	(Pointer to Occupa-tion)	Blacksmith	Or characteristic “hair”, value “red”.
S3b	Group	(Pointer)	Character-istic	(Pointer to Religious Habits)	Prays on Saturday	
S4a	Persona	(Pointer)	Persona	(Pointer)	Before	One name used before another.
S4b	Event	(Pointer to wedding)	Event	(Pointer to birth)	Before	
S4c	Character-istic	(Pointer to teamster)	Character-istic	(Pointer to black-smith)	Before	
S5	Persona	(Pointer to William Smith)	Group	(Pointer to children of union of Robert and Mary)	3rd child	
S5	Group	(Pointer to group of 7 families in Montgom-ery Co, NY)	Group	(Pointer to group of 7 families in Portage Co, Ohio)	Equal	

The table on the following page shows some example data using the super statement form above.

Subject 1 Type	***Subject 1 ID (Pointer to:)***	***Subject 2 Type***	***Subject 2 ID (Pointer to:)***	***Value (Pointer to:)***
Persona	George Smith	Event	Marriage of John Smith and Mary Jones	Father of the Groom
Persona	George Smith	Character-istic	Occupation	Blacksmith
Persona	George Smith	Group	Group of Personas #117 to be merged	
Event	Sherman Oaks Road Built	Event	Flood of 1877	After
Character-istic	Butcher	Group	Occupations that John Smith held	2
Character-istic	Teamster	Group	Occupations that John Smith held	1
Character-istic	Nick Name: Butch	Group	Names that John Smith was called by during his life	1
Character-istic	Nick Name: Skip	Group	Names that John Smith was called by during his life	2
Character-istic	Name: Mcaskell	Group	Name variations of M. Haskell	Equivalent
Character-istic	Name: M. Haskell	Group	Name variations of M. Haskell	Equivalent
Group	Passengers on the Titanic	Event	Sinking of the Titanic	
Group	Group of Personas #117 to be merged	Persona	George Edward Smith	

4.0 SCOPE AND REQUIREMENTS

4.1 SCOPE

Before we can discuss a genealogical data model, it's necessary to define the boundaries of what we mean by genealogical data, and if the subject matter is too large to tackle at one time, which it is here, to define the reduced boundaries in which we will work. The scope of activity for this data model was defined to include the following.

ADMINISTRATION

- Research Plan: keeping track of the research projects
- Research Assignment: a subset of a research project that outlines the purpose for a particular site visit or similar research endeavor
- Researcher and submissions information: an audit trail for mixed data entry. This concept supports the exchange of data electronically, so that the data, while appearing to become part of the researcher's database, can still be identified as to origin, and can be selected, changed, or removed as needed.
- Merging audit trail: the origin of data is never lost during merges.
- Notification: when data changes that has already been extracted or exported from the database and sent to someone, this administrative function allows updates to be sent.

It should be noted that research assignment and notification are particularly important for group efforts such as one-name studies.

EVIDENCE

- Evidence: full or selective data capture
- References: citations, sources, and repositories

CONCLUSIONS

- Analysis and commentary, also called investigative transformation: what the researcher does to the data including not only the transformed data but the rationale for the transformation as well

OUTPUT

- Compilation: family groups, pedigree charts, and other output formats

Note that these areas of scope can be seen clearly in the complete data model in Section 5.1 THE ENTIRE GENEALOGICAL DATA MODEL on page 23 and are used to section parts of the model. The only exception is Output, which is a function, and does not map to a data storage submodel.

We recognized the need to extend the data model to the following areas, but *did not* include these areas in the initial scope of the model because they form a logical extension of the core materials. We felt the smaller scope was a sufficient challenge at this time.

EXPERT ASSISTANCE

- Generally consists of rules for generating values, and these rules are culturally based. Expert systems require databases of knowledge in the form of rule sets, and for genealogical data many of these rules would be highly culture-dependent.

- Person Name knowledge. Useful for understanding variations of the same person name, or transformations of the same person name across cultural boundaries.

- Place knowledge. Allows the researcher to understand the boundary conditions of places, synonymous place names in the same or different languages, hierarchical place divisions, and overlapping jurisdictions associated with place data.

- Date and calendar system knowledge. Allows the researcher to make relevant date comparisons between different systems, to calculate time duration, and to utilize a knowledge of calendar systems to make other genealogically relevant conclusions.

  Note that we discuss possible expert system additions to the data model in Section APPENDIX C: DATA MODEL CONNECTIONS FOR EXPERT SYSTEMS on page 94.

DATA TRANSFER

- Although the main purpose of the data model was to facilitate an understanding of genealogical data, and one of the purposes for *that* was to facilitate electronic data transfer, the specification of data transfer is outside the scope of this initial effort.

DATA VALUES

- No data values, such as a list of events, were created as part of this data model.

4.2 REQUIREMENTS

Although the focus of this project is to create a data model that will explain how genealogical data is related, there are certain ***requirements*** related to genealogical research that strongly influence the definition of various genealogical data elements. These requirements help focus the reader on what sort of system this data model would support, and may be useful in helping understand the direction taken in some parts of the model.

4.2.1 Basic Research Requirements

The data model must support the strongest reference and citation system possible, tracking all assertions directly to the evidential data.

The data model must support the concept that each piece of data pertains to a different person, even if of the same name, until proven otherwise.

The data model must allow an implementation suitable for novices in addition to experienced researchers, without compromising the integrity of the data. In practice, this means that an actual implementation would automatically fill in missing attribution data as needed with place holders such as "Unsupported" or "Undocumented".

4.2.2 Person Name Requirements

The decomposition of person name into name parts is discussed in the section C.1 PERSON NAME EXPERT SYSTEM on page 94. Note that many of the requirements, however, do not in any way suggest the use of name parts.

- Names must be stored exactly as found.

- Names must be available to the genealogical researcher as a single, meaningful chunk of data.

- Each different name *begins* as a different Persona-ID in the PERSONA entity. Thus, one thinks of a real person and the different personas that person had over his or her lifetime. Each persona has associated with it a number of genealogical records. The various personas of a single person can be associated to form a composite individual over time, a Constructed or Master Persona. (We have simplified this concept in the model to various "Low" personas that are combined into a "High" persona, consistent with the way we named other attributes.) A description is necessary to name a "Constructed Persona" for purposes of display on charts that require a single name.

- To properly sort, sequence, and index names requires an expert system. A future requirement is that the expert system accepts as input a name with an associated place and date, and returns the name in the correct sort order, and with additional name entries for indexing purposes. In addition, the expert system returns the name in pieces according to a statistical profile that suggests how likely the pieces are to be correctly identified, a statement of the probable cultural identity of the name or name part, and the standard spelling for that and related names.

4.2.3 Place Name Requirements

Place Names are discussed in Section C.2 PLACE NAME EXPERT SYSTEM on page 95. The following are the requirements for Place Names.

- A place name must be captured in its entirety, exactly as found.
- Each place name or variation is a different Place-ID.
- Place names must allow imprecise locations such as "near".
- Place names must be sorted and indexed according to rule sets related to the hierarchy of place names.
- A future requirement is that an expert system must be able to identify a place name as to spatial coordinates on a fixed grid, and must recognize the time period that that place name applied to. In other words, if a particular place such as a county (or an entire country) had different boundaries at different times, the expert system recognizes those boundaries associated with each time period. The user may be required to keep the name parts in either ascending or descending order, either as explicitly stated or through default, as input to the expert system.

4.2.4 Date Requirements

Dates are discussed in Section C.3 DATE EXPERT SYSTEM on page 96. The group identified the following requirements related to dates.

- Capture the actual date exactly as found in the original record, no matter how unusual.
- Identify the calendar and the place of the date, e.g., Gregorian-U.S.
- Allow soft (approximate) dates.
- Allow ranges.
- Allow relative dates.
- Any entity that contains date information should have associated with it a user-definable sort date to allow the data to be sorted as appropriate.
- The system must allow the user to specify a rule set related to the sorting of soft dates, such as dates with Month and Year only are to be sorted as the 1st of the month (or the last of the month, or the 15th of the month).
- The formatting of dates for output should be user selectable, so that all dates, regardless of how they were entered, can be converted to a single base format, or not converted as the user wishes.

- The future requirement for a date expert system is that the system submits the date, the place, and the calendar system to the expert system, and the expert system returns the date in Julian Period format.[4]

4.2.5 Attribution and Administrative Requirements

- The data model must allow all evidential and conclusional data to be tracked to the researcher who originated it.
- The data model must track all data exported from the system to other researchers.
- The data model must support single researchers, multiple researchers cooperating in projects like one-name studies, and professional researchers who have multiple clients but who wish to share some data across client projects.

[4] The Julian Period number should not be confused with the Julian Calendar. The Julian Period number is an integer value representing a single day. Thus, a date in one system can be converted to a date in another system directly by knowing the relationship, or dates can be freely converted between multiple calendar systems by converting the input date to a known, unambiguous, numerically useful standard like the Julian Period date, and then converting that standard back out to the target calendar system. Most (but not all) Commercial Off the Shelf Software (COTS) such as Excel, Access, and so forth uses some variation of the Julian Period date to store dates internally. That's why Excel, for example, has no difficulty adding, say, 22 days to a given date; it's simply an integer addition to the Julian Period integer representation of the date, with the answer translated back to a Gregorian Calendar date for the user to read.

5.0 GENTECH GENEALOGICAL DATA MODEL

There are a number of ways the entire logical data model can be subdivided, but one of the most critical parts is the relationship of conclusional data, data such as a birth date that might appear on a particular pedigree chart for an individual, and the evidential data that was first collected that led to that conclusion. The storage and manipulation of evidential data is the most significant difference between this data model and most de facto data models in use today in genealogical software.

5.1 THE ENTIRE GENEALOGICAL DATA MODEL

The entire data model appears in the foldout after this page. The drawing and naming convention information appears below.

5.1.1 Naming Conventions

The following are the naming conventions used in the data model.

- All entities are named in the singular, such as SOURCE instead of SOURCES.

- The name of an entity is capitalized throughout this document, for example RESEARCHER instead of Researcher.

- When speaking of entities and relationships, the plural is formed from an entity name by appending a lower case "s", not by forming the correct English plural. This is because the entity has a specific name, and we don't want to change it just for grammatical reasons. Thus, we say that one RESEARCHER works on many PROJECTs, but we also say one SOURCE exists in zero to many REPOSITORYs.

- Attributes of an entity are named in mixed case, such as Researcher-ID.

- The names of attributes as well as the names of entities are hyphenated when more than one word is used so that the reader understands one complex name is meant, such as DATA-RECIPIENT, and Data-Recipient-ID.

- Associative entities, because they are somewhat abstract creations designed to break up many to many relationships, are arbitrarily named from the entity on the left plus the entity on the right, as the model is currently drawn. If the two entities are above and below, the associative entity is named from top to bottom.

- Independent entities that do not require other entities are drawn with square corners.

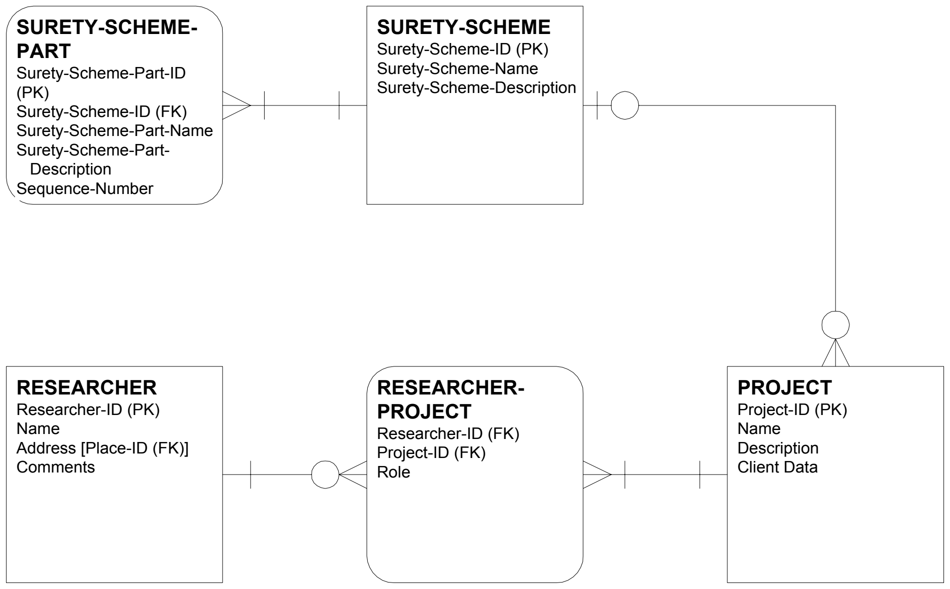
- Dependent entities that rely on other entities are drawn with rounded corners.

5.1.2 Connectors

The legend on the data model shows the various connectors and how to read them, but it should be noted that there is no significance to where they connect on a given entity. They are not designed to connect one listed attribute to a similar attribute in another entity.

5.2 ADMINISTRATION SUBMODEL

The Administration Submodel refers to the entities primarily used to keep track of the project, such as the researcher, project, and research objectives, and the searches and administrative tasks conducted or planned. Part of the Administration submodel appears below.

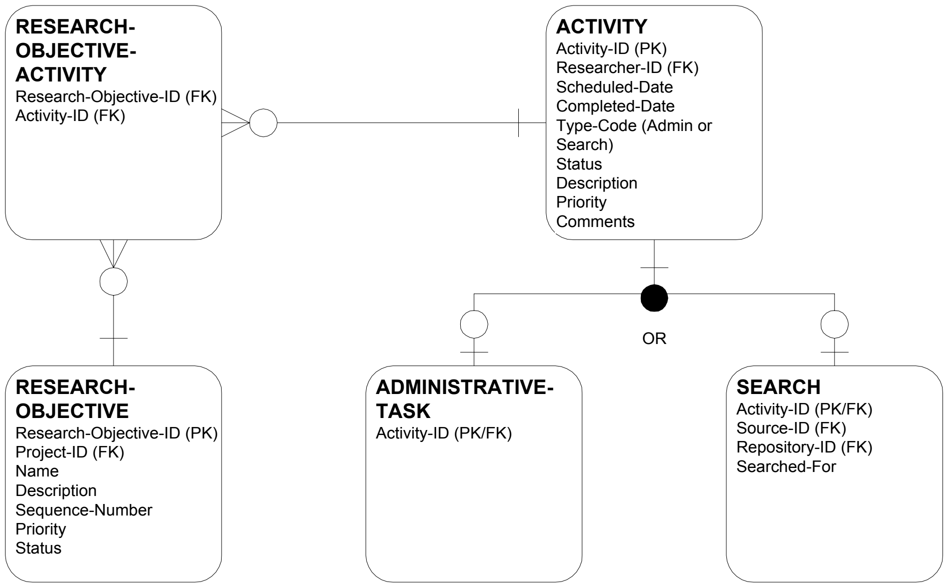


The RESEARCHER entity allows genealogical data to be linked to the particular person who gathered that data, whether raw evidence or conclusions based on evidence (or based on nothing at all in the case of poor research). Although the RESEARCHER entity is very useful for group efforts such as one-name studies, it is also necessary so that data that enters or leaves the system can be attributed to a researcher.

The PROJECT entity identifies one or more projects that the researcher is interested in. It is linked to the RESEARCHER entity with an associative RESEARCHER-PROJECT entity required because one researcher may work on zero to many projects and one project can have zero to many researchers. After data becomes available, of course, a project must have at least one researcher. We will not discuss associative entities throughout the rest of the model unless there is something particularly interesting about them. The reader is referred to APPENDIX A: PRINCIPLES OF LOGICAL DATA MODELING for a further discussion of associative entities used to break up a many to many relationship between two entities. Note that not all dependent entities (shown with rounded corners) are merely used for this purpose; many are significant in their own right, but cannot contain data without first having data in other independent entities.

Every PROJECT has associated with it a SURETY scheme used to express how certain the researcher is of the data gathered. Although many people use the same scheme, as implemented in certain software or elsewhere, the data model allows different schemes to be used. Note, however, that the scheme used with a particular project is attached to the data that it describes; this means that an export specification[5] can send the surety scheme along with the data.

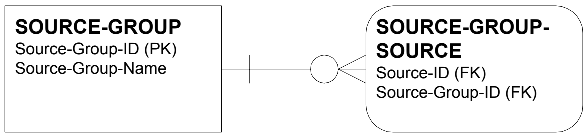
PROJECT is associated with zero to many RESEARCH-OBJECTIVEs as shown in the rest of the Administration Submodel below.



Researchers can specify their goals in RESEARCH-OBJECTIVE. The data in RESEARCH-OBJECTIVE explains what research problem the genealogist is attempting to solve.

The RESEARCH-OBJECTIVEs are linked to specific ACTIVITYs such as a SEARCH or an ADMIN-TASK, planned or already executed. A SEARCH takes place at a particular repository, on a date, using a SOURCE (like a will book), and consists of examining the SOURCE for a particular person name or names, a particular place name or names, or other data. Thus, the SEARCH is the end of this particular chain of entities in the Administration Submodel, and links to the Evidence Submodel.

There are two final entities in the Administration submodel that support SOURCE in the Evidence Submodel.



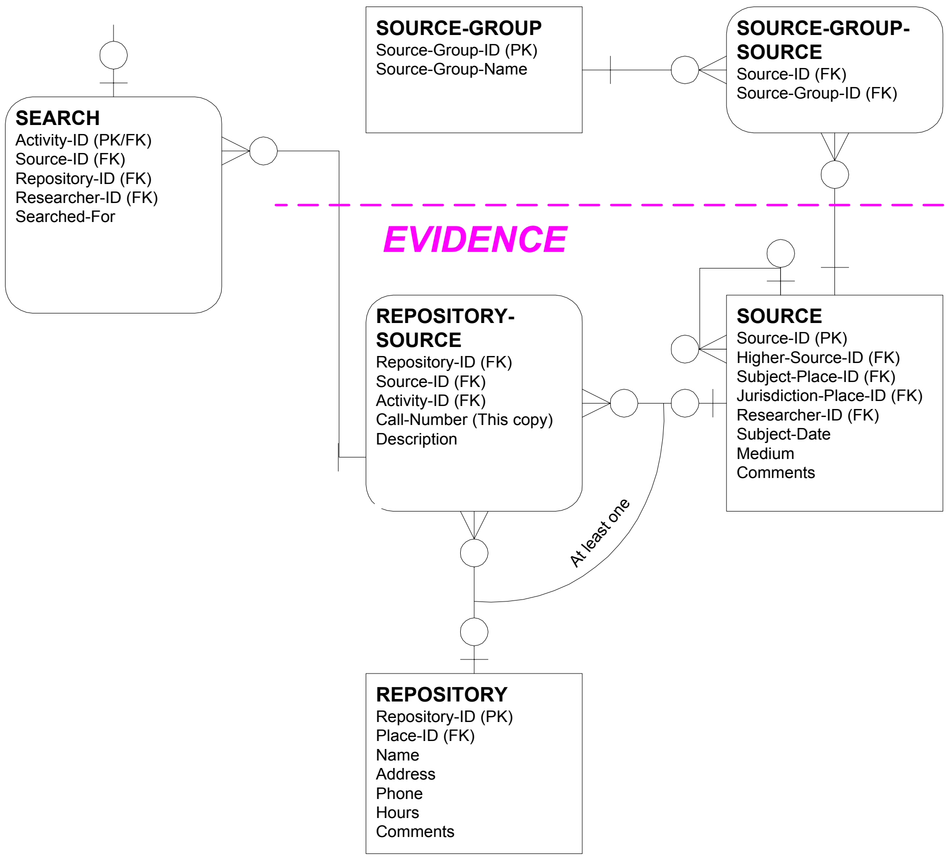
These two entities allow SOURCEs to be grouped however the RESEARCHER would like. At the higher levels of SOURCE, for example, one might want to create a SOURCE-GROUP which includes all those sources which one would search for a certain county in which one frequently works. At a lower level in the SOURCE hierarchy, one might create a SOURCE-

[5] It should be noted that this data model does not have an export specification, however.

GROUP for a set of records which are related in some way, perhaps a series of tax lists for one town or county.

5.3 EVIDENCE SUBMODEL

SEARCH, from the Administration Submodel links into the Evidence Submodel by connecting to REPOSITORY-SOURCE, an associative entity that connects both REPOSITORY and SOURCE.



The REPOSITORY entity contains data that identifies the repository such as name and address. Note that a repository need not be a public library or archives; the entity can contain data about an individual who provided genealogical information. Conceptually, this individual *is* the repository.

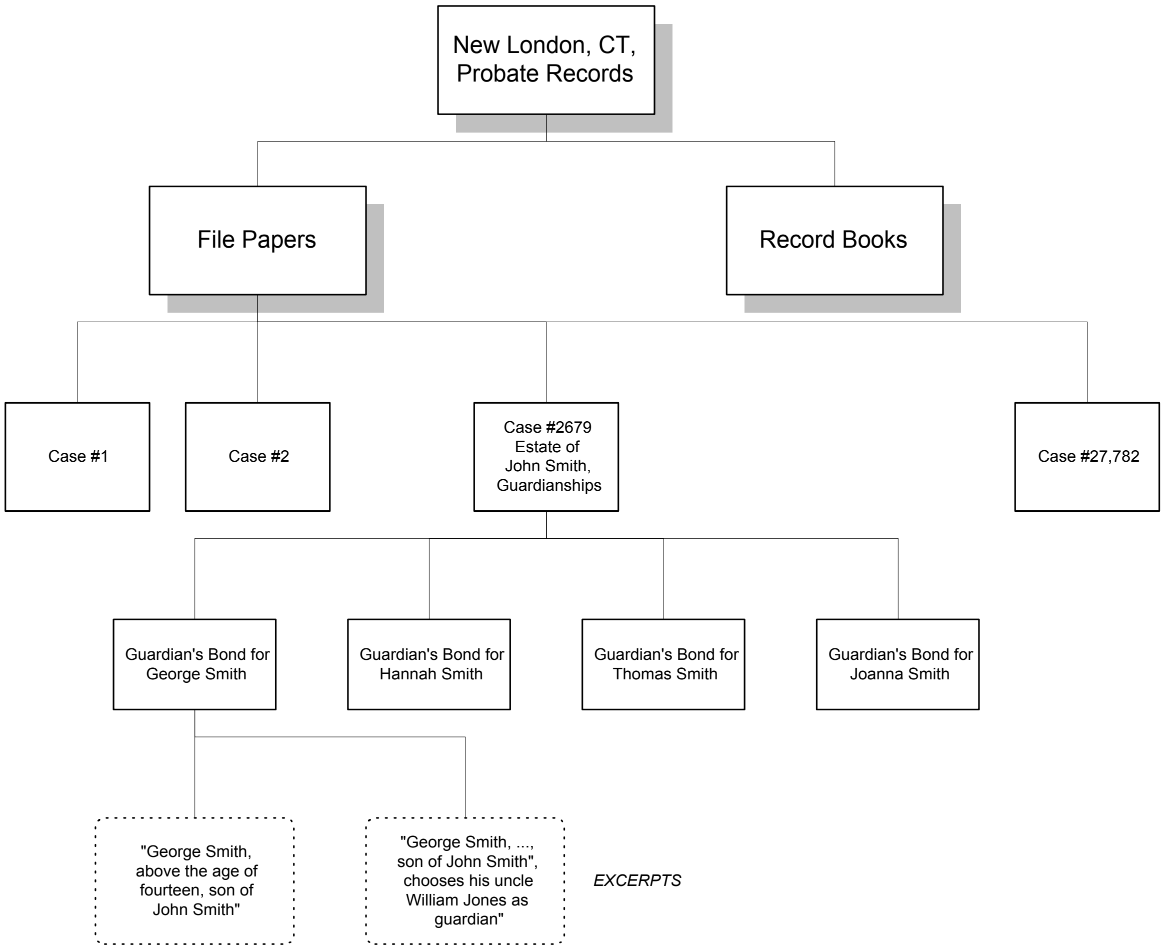
Consider the relationship between REPOSITORY and SOURCE. Clearly, one REPOSITORY has potentially many SOURCEs, meaning that if we go to the National Archives, an instance of REPOSITORY, we will potentially find many census records, instances of SOURCE. On the other hand, a particular SOURCE (e.g., a microfilm reel with the Prince George's County, MD 1870 federal census on it) will be found in one to many REPOSITORYs. It must be in at least one repository, but it may be in more.

An exception occurs, however, when the researcher is aware of a particular SOURCE, but does not know where to find it. Often this is a primary or secondary work mentioned in a secondary source; the existence is known, but not the physical whereabouts. Consequently, the model indicates that one SOURCE is found in zero to many REPOSITORYs and not one to many REPOSITORYs as might otherwise be supposed.

SOURCE-REPOSITORY is a convenient place to hold the call number for a SOURCE because the call number for the same source may be different at different REPOSITORYs.

The notation "At least one" indicates that every instance of REPOSITORY-SOURCE must be connected to at least SOURCE or REPOSITORY or both. In a normal SEARCH it will be connected to both.

Note that SOURCE is self-referential. Each high level SOURCE may have zero to many lower level SOURCEs. This concept is best seen in the following chart showing a hierarchy of SOURCEs in the records of one particular court.


New London, CT, Probate Records
File Papers
Record Books
Case #1
Case #2
Case #2679 Estate of John Smith, Guardianships
Case #27,782
Guardian's Bond for George Smith
Guardian's Bond for Hannah Smith
Guardian's Bond for Thomas Smith
Guardian's Bond for Joanna Smith
"George Smith, above the age of fourteen, son of John Smith"
"George Smith, ..., son of John Smith", chooses his uncle William Jones as guardian"
EXCERPTS

Initially, it is tempting to break SOURCE up into perhaps *DOCUMENT* and *RECORD* or some similar scheme until the actual *EXCERPT* level is reached.[6] However, an example like this one quickly shows that we cannot count on a particular number of levels in SOURCE for all records. This example has four levels above the excerpt level; many examples with fewer

[6] We mention *DOCUMENT*, *RECORD*, and *EXCERPT* in this paragraph as if they were entities in a particular scheme to illustrate the concept of a discrete division between levels of SOURCE. In the current data model these entities do not exist.

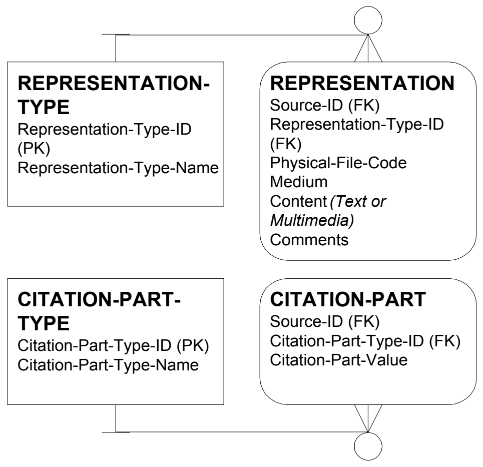
levels come to mind, and there are probably examples with even more levels. Thus, the only logical way to model SOURCE data is as a self-referential hierarchy with an unknown number of levels.

SOURCEs include both primary and secondary works. There are a large number of possible genealogical SOURCEs such as will books, deed books, compiled genealogies in book or periodical form, and electronic databases. If there are multiple copies of a SOURCE, break them out at the lower level of the SOURCE hierarchy, and draw the ASSERTION from that level or lower.

If a SEARCH is a repository-wide search for particular types of records, for example, there will be zero SOURCEs associated with the SEARCH. If the SEARCH takes place in a particular document, there will be one SEARCH for one SOURCE.

In reversing this situation, however, it is possible to note a SOURCE without actually searching it, or we might search for one or more items in the source. Consequently one SOURCE has zero to many SEARCHs, and one SEARCH has zero to one SOURCE.

We also have a SOURCE-GROUP entity (in the Administration submodel) that allows us to group different kinds of records together, so that we can search our database for records from certain kinds of sources, such as only wills, or only census records.



The SOURCE entity contains none of the information required in the citation such as the title and author, or any of the other "book-level" data associated with the many kinds of sources.

The citation associated with each level of SOURCE is stored in CITATION-PART. For example, Citation-Part-Value holds "New London, CT Probate Records" as the top level CITATION-PART. The type of citation that this information represents is stored in CITATION-PART-TYPE; in this example it might be "Probate Jurisdiction".

The actual content of the citation is stored in REPRESENTATION. This is where the excerpts from our New London example are stored. In addition, REPRESENTATION can hold multi-media content such as a voice recording or an electronic image. The type of REPRESENTATION, such as "Bitmapped Image" or perhaps "TIF File" in a different scheme, is referenced in REPRESENTATION-TYPE.

Note that the examples discussed above are of a REPRESENTATION *in* a SOURCE, meaning a text excerpt, an image of part of the SOURCE, and so forth. However, it is also possible to have a REPRESENTATION *of* a SOURCE, meaning a "picture" of the SOURCE. The difference between "in" and "of" is relatively subtle, but the model accommodates both.

An observation should be made at this point about the relationship between evidential data and multiple projects. A professional genealogist, for example, will gather evidence for a particular PROJECT, but at a later time may find some of that same evidence useful for another client and another PROJECT. The data model does not demonstrate a direct connection between SOURCE and PROJECT, so we cannot look at the model and say, for example, a SOURCE is used for one (or perhaps zero) to many PROJECTs.

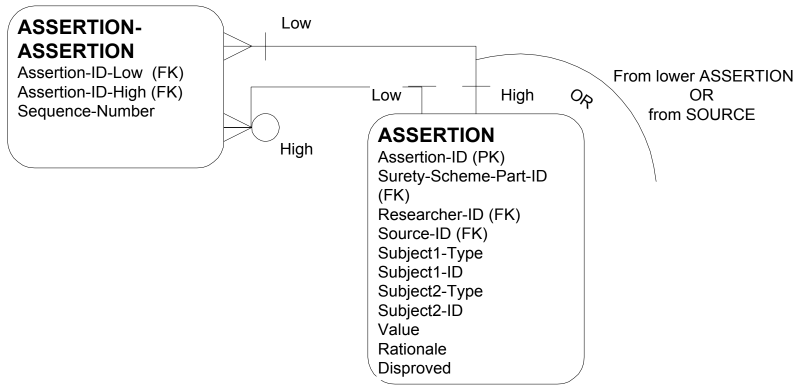
In order to understand whether our model in fact supports reusing evidence in this way, it is necessary to examine the series of links *between* SOURCE and PROJECT. Assuming that a particular SOURCE was used in support of a particular RESEARCH-OBJECTIVE (remember that some SOURCEs exist independently of any SEARCHs or RESEARCH OBJECTIVEs), then we know that one SOURCE supported at least one (not zero) SEARCHs, and that SEARCH supports one and only one RESEARCH-OBJECTIVE, and that RESEARCH-OBJECTIVE supports one and only one PROJECT.

But the key here is that the professional genealogist, in this example, can start with another PROJECT that creates another RESEARCH-OBJECTIVE that creates another SEARCH that happens in the same SOURCE. Thus, the relationship between one SOURCE and zero to many SEARCHs allows the researcher to reuse SOURCE data for multiple projects.

Using this same chained reasoning, it is possible to examine other kinds of situations to determine whether the data model fits all known research conditions. When examining a chain, if all the links in one direction have at least a one (if not one to many) condition on the far side of the link, then a relationship holds across the chain. If any condition in the chain has a zero condition, the chain will be broken by certain data and thus a relationship will *not* always exist. This is why there are links from many entities back to RESEARCHER, for example—we cannot rely on a chain of data back that far.

5.4 CONCLUSIONAL SUBMODEL

The heart of the data model is the Conclusional Submodel, specifically the critical role that the ASSERTION entity plays. Once a piece of evidence has been isolated down to the atomic level of a simple excerpt from the SOURCE, the RESEARCHER then uses that statement to make an ASSERTION. Note that the actual excerpt is stored in REPRESENTATION, but it is available to the ASSERTION entity through a SOURCE record, specifically the lowest level SOURCE record in the hierarchy.



5.4.1 The General Concept of an ASSERTION

ASSERTIONs can be created in two ways.

- By converting a SOURCE fragment into an ASSERTION, placing it in the general statement form derived from our Super Statement form. Each SOURCE can give rise to one or more ASSERTIONs, but each ASSERTION created this way derives from one and only one SOURCE fragment.

- By making an ASSERTION based on one or more existing ASSERTIONs.

The power of the ASSERTION entity is that the fragments of evidence that are gathered in SOURCE are then used directly to build low level conclusions. Higher level conclusions are in turn built from lower level ASSERTIONs as needed, thus giving a complete audit trail that shows why an ASSERTION (a conclusion) was made.

The data model also accommodates a negative ASSERTION, a statement that something is *not* the case. An example of this is the following statement, "Cpl. Smith was wounded in a skirmish prior to the Battle of Gettysburg and did not participate in that fight." The discussion of this is somewhat complex; the usefulness of the concept to genealogical researchers depends on whether significant negative information is available. Many other parts of the model support negative information as well.

Once the general concept of an ASSERTION is understood, it's necessary to understand how all genealogical statement types are captured by this construct. These statements are broken into four subject types and discussed in the following section.

5.4.2 The Four Subject Types in ASSERTION

Each ASSERTION contains a statement in the following form. (See section 3.3 The Super Statement Type on page 15 for an explanation of this compact genealogical statement format.)

Subject1-Type / Subject1-ID / Subject2-Type / Subject2-ID / Value

The four subject types are PERSONA, EVENT, CHARACTERISTIC, and GROUP, and the ID points to a specific occurrence of that type. Thus, we can connect data in the following matrix. The cells that have a check mark in them indicate the most likely combinations, but the model does not prohibit any particular combination. *Proposed* prohibitions, however, or combinations with some restrictions are shown as grayed out cells, and are explained by rules in the appropriate sections below.

		Subject 2			
		PERSONA	EVENT	GROUP	CHARAC-TERISTIC
Subject 1	PERSONA	▨	✓	✓	✓
	EVENT	▨	✓	✓	
	GROUP	✓	▨	✓	▨
	CHARACTERISTIC	▨	▨	✓	▨

The top row, for example, shows subject 1, PERSONA connected to the following.

- PERSONA: a relationship to another PERSONA (person 1 is the father of person 2). Note, however, that although this is the classic S1 statement type, we later suggest that a stronger way to capture relationships is through the GROUP construct. See section 5.4.6 Persona Entities on page 40.
- EVENT: an event in the life of the PERSONA (person was the groom at a wedding)
- GROUP: the PERSONA is a member of the GROUP (person was a sergeant in a particular troop)
- CHARACTERISTIC: the PERSONA has a certain CHARACTERISTIC (the person had red hair)

The general case and the specific case for each of the examples in the table above are shown below.

Subject1-Type / Subject1-ID / Subject2-Type / Subject2-ID / Value

PERSONA / 1234 (William Smith) / PERSONA / 4321 (Sarah Smith) / Father[7]
PERSONA / 1234 (William Smith) / EVENT / 2222 (Smith-Jones wedding) /Groom
PERSONA / 1234 (William Smith) / GROUP / 3333 (F Troop) / Sergeant
PERSONA / 1234 (William Smith) / CHARACTERISTIC / 4422 (Hair color) / Red

The cells in the table that are checked are those for which it is relatively easy to construct a statement type such as GROUP to EVENT (Seven Montgomery County, NY families moved to Ohio) or GROUP to GROUP (the group of 7 families from New York are part of the group of 51 families that appear a decade later in Ohio).

The cells that are not checked are more difficult to populate with meaningful examples, but the data model does not prohibit constructing those types of ASSERTIONs except through several rules presented in sections 5.4.5 Group Entities and 5.4.6 Persona Entities on pages 38 and 40. The four types of subjects are discussed in subsequent sections below.

Each ASSERTION relies on either a SOURCE for its underlying evidence, or on one or more lower level ASSERTIONs. ASSERTIONS can be built up as needed into increasingly higher level ASSERTIONs. ASSERTIONs that are later disproved *could* be purged from the data, but a better method is to mark them as disproved; this preserves the chain of reasoning that led to the later-proved-erroneous conclusion to prevent the RESEARCHER from making the same mistake years later. Each ASSERTION, of course, also contains the Rationale used by the RESEARCHER to make that ASSERTION.

The large, solid dot on the relationship connector below ASSERTION indicates that it is an “OR-gate”. An OR-gate connects an entity to one of a series of subtypes. In this case, however, it indicates that ASSERTION is connected to *two* of the possible four types, corresponding with the Subject1-Type and Subject2-Type elements of the Super Statement Type.

The following page contains three more examples of paired entities in an ASSERTION using more complete sample data. In this case, the three examples are related.

[7] See the discussion in section 5.4.6 Persona Entities on page 40. This S1 statement type is no longer used, but is presented for continuity with the earlier discussion.

Sample ASSERTION: PERSONA to CHARACTERISTIC

Assertion: Charles G. Ferris had liver disease in 1877.

Rationale: His brother, a surgeon, said so in a sworn statement.

Evidence: “His [sic] is Dyspeptic, and has obstinate functional derangement of the liver.” Statement of Surgeon Orrin Ferris. Declaration for Original Pension of Charles G. Ferris, #245,225, 1877.

Sample ASSERTION: PERSONA to GROUP

Assertion: Charles Ferris was in the 123d Ohio Volunteer Infantry.

Rationale: He is listed as such in his accepted pension application.

Evidence: “Personally came Charles G. Ferris… who was a Chaplain in the 123d Regiment of Ohio Volunteers war of 1861…” Declaration for Original Pension of Charles G. Ferris, #245,225, 1877.

Sample ASSERTION: GROUP to GROUP

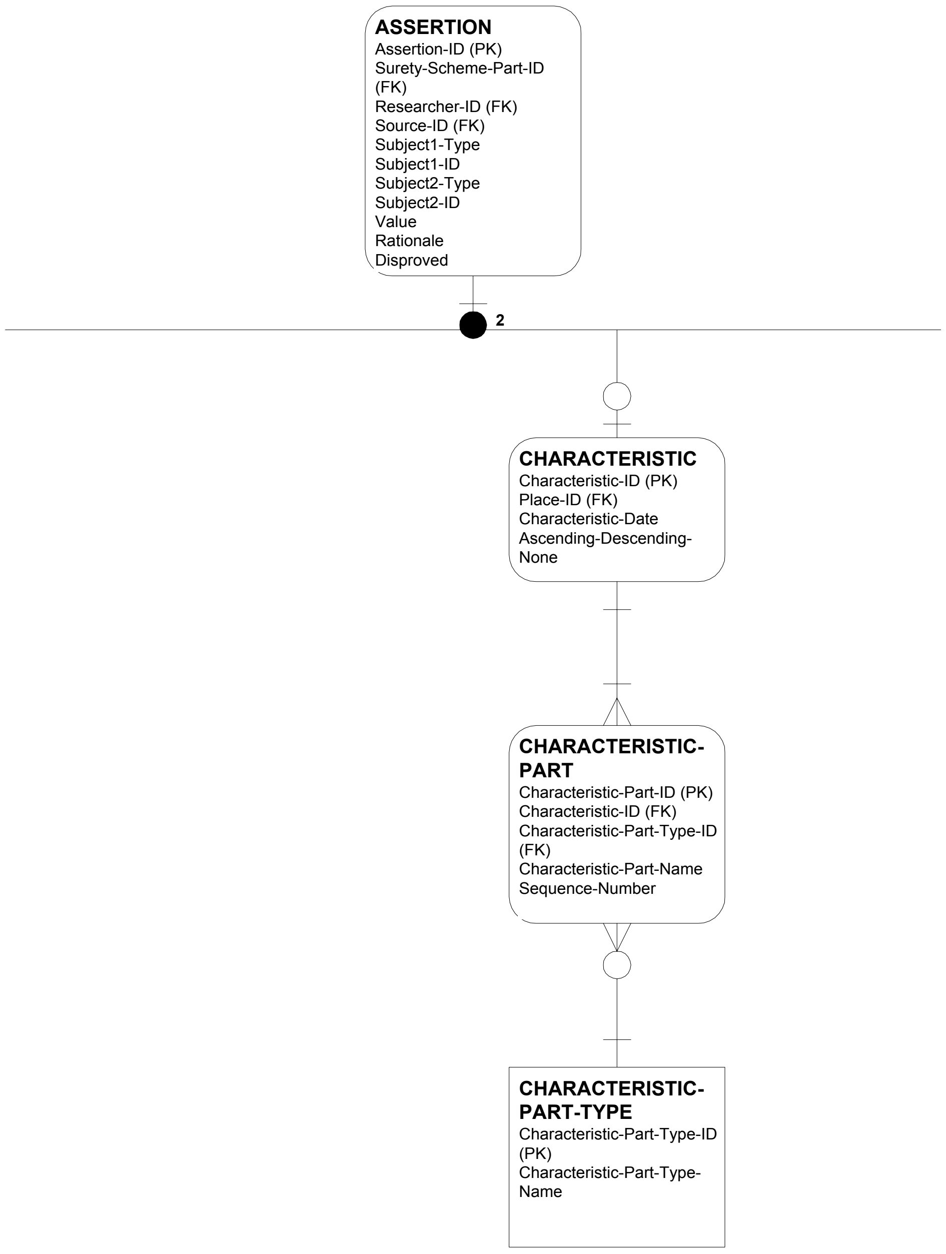
Assertion: Some 123d Ohio Volunteer Infantry (OVI) officers served at Taylor Army Hospital in the 1st Battle of Winchester.

Rationale: Surgeon Orrin Ferris and Chaplain Charles G Ferris, known to belong to 123d OVI are reported at Taylor Army Hospital in June 1863 in an accepted pension application.

Evidence: “Personally came Charles G. Ferris… who was a Chaplain in the 123d Regiment of Ohio Volunteers war of 1861… In June 1863… while waiting on the sick and wounded soldiers in the camps in the Shanandoah Valley… and particularly at Taylor Hospital Winchester VA… was dragged… by the revel officers and thrown… among the sick and wounded…” Declaration for Original Pension of Charles G. Ferris, #245,225, 1877.

5.4.3 Characteristic Entities

When one of the attachments, either subject 1 or subject 2, describes a characteristic, it is connected to CHARACTERISTIC. Unlike the group or event series of entities, the value in ASSERTION is not connected directly to CHARACTERISTIC-PART which holds the values that we would expect to find directly pointed to. Instead, the Value attribute in ASSERTION is connected to the Characteristic-ID in CHARACTERISTIC so that we can use the characteristic entities to hold not only characteristics like specific occupations, but also person names. Because of this, there is an additional attribute called Ascending-Descending-None that is used to order characteristic parts, chiefly names.



If the ASSERTION is about a characteristic like a person's name, there will be one CHARACTERISTIC for one to many CHARACTERISTIC-PARTs. If the characteristic is occupation, for example, there will be one CHARACTERISTIC-PART.

Each CHARACTERISTIC-PART is of one and only one CHARACTERISTIC-PART-TYPE. For example, the Characteristic-Part-Name "Sitting Bull" is of the Characteristic-Part-Type-Name "Mononame", while "John" is a "Given Name", "Cardinal" is an "Infix", and "Smith" is a "Surname" (in "John Cardinal Smith"). The Characteristic-Part-Name "Blacksmith" is the Characteristic-Part-Type-Name "Occupation". Of course a CHARACTERISTIC-PART-TYPE has zero to many CHARACTERISTIC-PARTs. For example, there are zero to many occupation names, and there are zero to many medical conditions such as "Obstinate Liver".

The Lexicon Group, while not prohibiting all combinations of cells in the table of subject types on page 30, nevertheless believes that that an appropriate ***rule*** about CHARACTERISTIC is the following.

- CHARACTERISTIC can only be Subject1 when GROUP is Subject2. This allows CHARACTERISTICs to be grouped where appropriate but removes other combinations that do not make any sense.

The chart below shows the possible values for attributes in the characteristic entities, where the ASSERTION is about an occupation.

ASSERTION	
(Type S3a)	
Subject1-Type =	Persona
Subject1-ID =	Persona-ID
Subject2-Type =	Characteristic
Subject2-ID =	Characteristic-ID
Value =	---
CHARACTERISTIC	
Characteristic-ID =	Match ASSERTION Subject
CHARACTERISTIC-PART	
Characteristic-ID =	Match CHARACTERISTIC
Characteristic-Part-Name =	Blacksmith
Characteristic-Part-Type-ID =	Match CHARACTERISTIC-PART-TYPE
CHARACTERISTIC-PART-TYPE	
Characteristic-Part-Type-Name =	Occupation

The chart on the following page shows sample values for attributes in the characteristic entities, where the ASSERTION is about a name.

ASSERTION

S3a

Subject1-Type =	Persona
Subject1-ID =	Persona-ID
Subject2-Type =	Characteristic
Subject2-ID =	Characteristic-ID
Value =	---

CHARACTERISTIC

Characteristic-ID =	Match ASSERTION Subject
A-D-None =	Ascending

CHARACTERISTIC-PART

Record 1

Characteristic-ID =	Match CHARACTERISTIC
Characteristic-Part-Name =	John
Characteristic-Part-Type-ID =	Match CHARACTERISTIC-PART-TYPE
Sequence-Number =	1

Record 2

Characteristic-ID =	Match CHARACTERISTIC
Characteristic-Part-Name =	Quincy
Characteristic-Part-Type-ID =	Match CHARACTERISTIC-PART-TYPE
Sequence-Number =	2

Record 3

Characteristic-ID =	Match CHARACTERISTIC
Characteristic-Part-Name =	Smith
Characteristic-Part-Type-ID =	Match CHARACTERISTIC-PART-TYPE
Sequence-Number =	3

Record 4

Characteristic-ID =	Match CHARACTERISTIC
Characteristic-Part-Name =	Esq.
Characteristic-Part-Type-ID =	Match CHARACTERISTIC-PART-TYPE
Sequence-Number =	4

CHARACTERISTIC-PART-TYPE

Record 1

Characteristic-Part-Type-Name =	Given Name

Record 2

Characteristic-Part-Type-Name =	Given Name

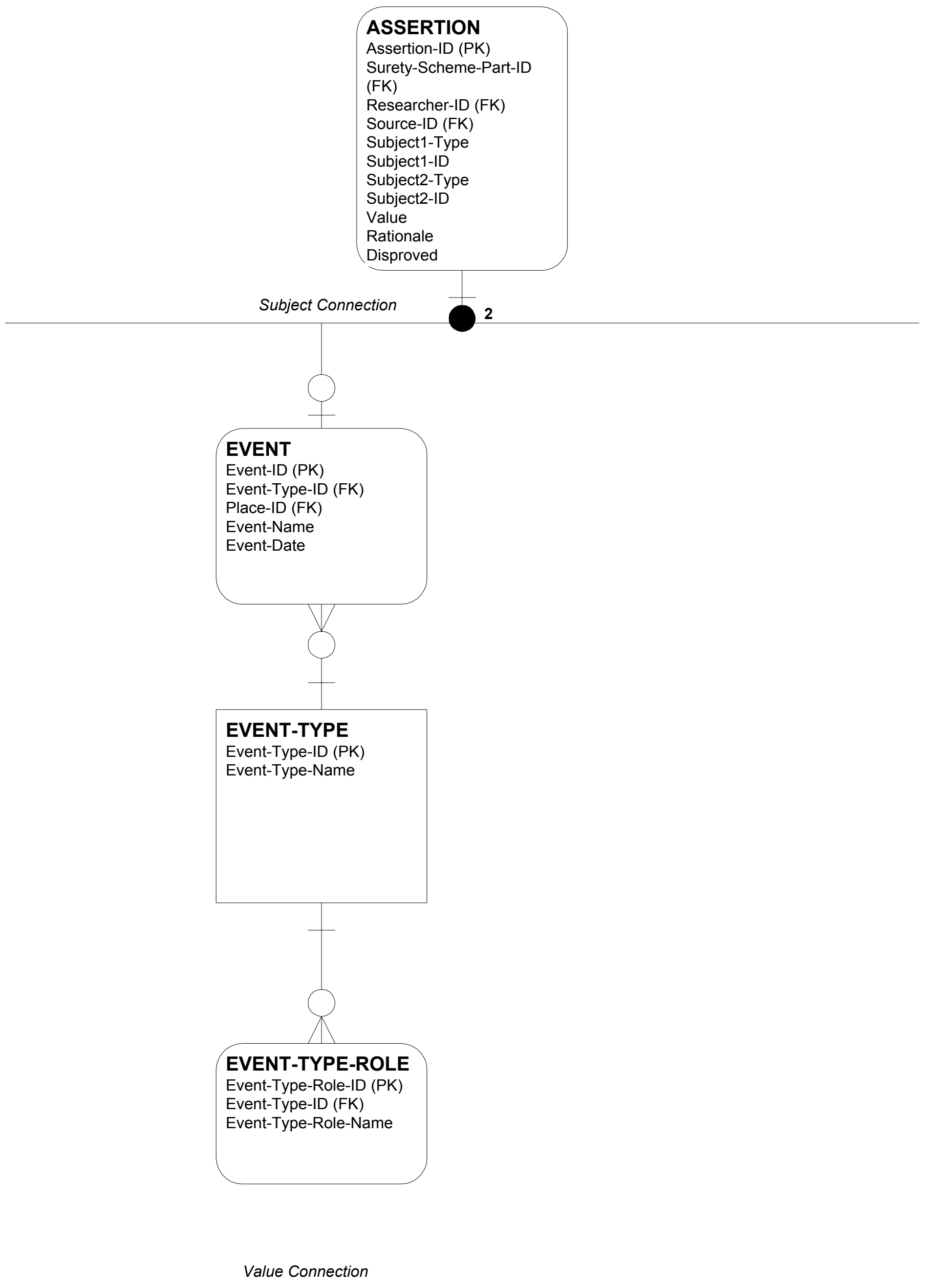
Record 3

Characteristic-Part-Type-Name =	Surname

Record 4

Characteristic-Part-Type-Name =	Suffix

5.4.4 Event Entities



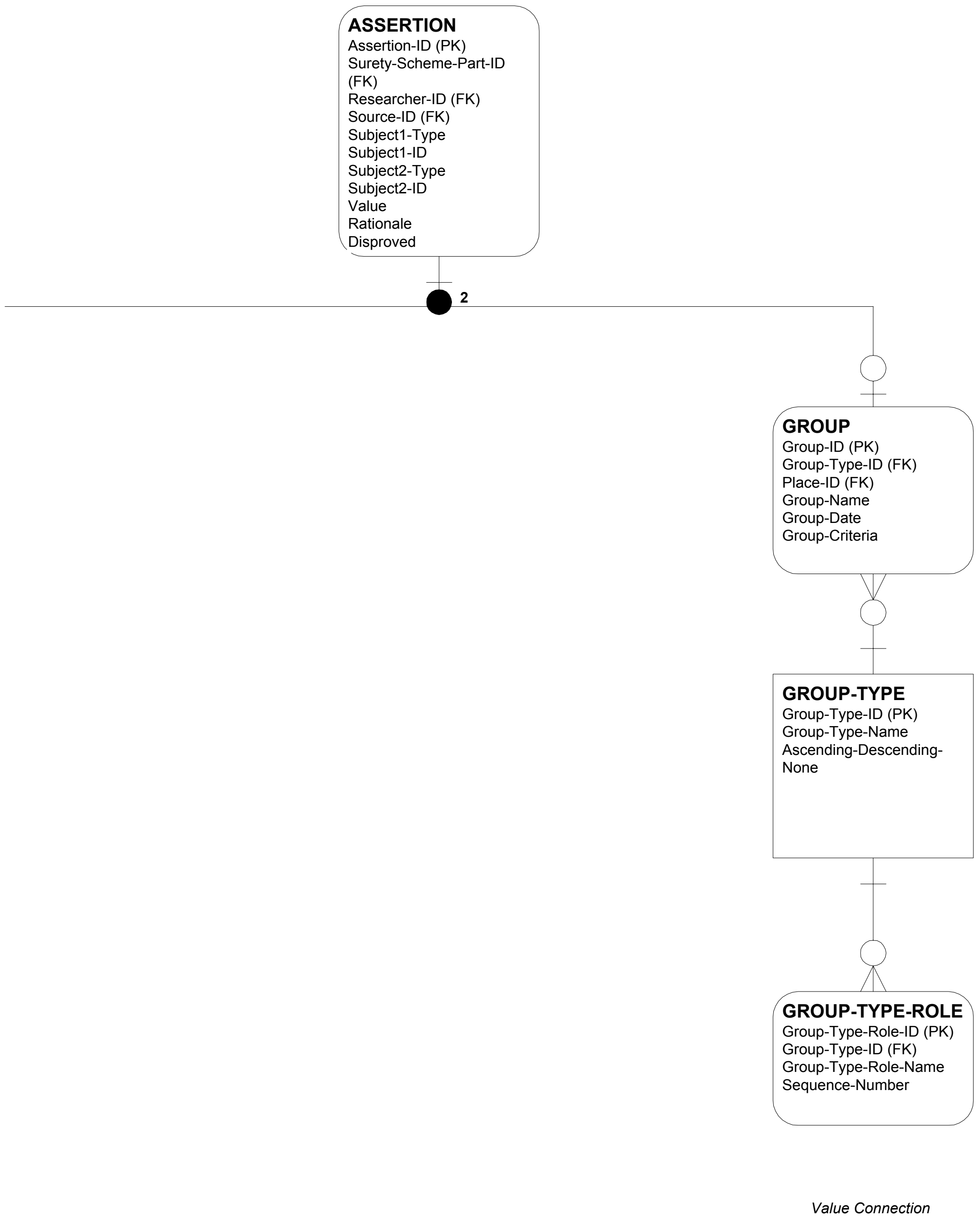
When one of the attachments to ASSSERTION, either Subject1 or Subject2, describes an event, it is connected to EVENT. An EVENT is a specific happening; Event-Name might be "Wedding of John Smith and Mary Brown". Each EVENT is of an EVENT-TYPE; in this case Event-Type-Name would be "Marriage". Clearly, for an EVENT-TYPE, there can be zero to many specific EVENTs.

While Subject1 or Subject2 are connected to EVENT, Value in ASSERTION is connected to EVENT-TYPE-ROLE. Value contains the Event-Type-Role-ID from EVENT-TYPE-ROLE. Depending on the Persona, the Event-Type-Role-Name that matches this ID might be "Groom". Since there is only one Value in ASSERTION, even if both Subject1 and Subject2 point to EVENT, there can be at most 1 Event-Type-Role-ID in Value, and often there is no Event-Type-Role-ID. Thus the connection is zero to one EVENT-TYPE-ROLEs.

The chart below shows possible values for selected attributes in the event entities.

ASSERTION	
Subject1-Type =	Persona
Subject1-ID =	Persona-ID
Subject2-Type =	Event
Subject2-ID =	Event-ID
Value =	Event-Type-Role-ID
EVENT	
Event-ID =	Match ASSERTION Subject
Event-Name =	Wedding of John Smith and Mary Jones
EVENT-TYPE	
Event-Type-Name =	Marriage
EVENT-TYPE-ROLE	
Event-Type-Role-ID=	Match ASSERTION Value
Event-Type-Role-Name =	Groom

5.4.5 Group Entities



When one of the attachments, either Subject1 or Subject2, describes a group, it is connected to GROUP. GROUP describes a particular collection of people or other objects.

Each GROUP is of a particular GROUP-TYPE. For example, an instance of GROUP-TYPE might be "Children of a Union", while a specific GROUP identified children of a particular union, such as the children of John Smith and Mary Jones.

One GROUP-TYPE is connected to zero to many GROUPs. For example, a Neighborhood GROUP-TYPE would presumably show up in any number of actual Neighborhood GROUPs. Each GROUP has a specific date and place; the GROUP-TYPE is the general category of GROUP.

But ASSERTION, when it applies to a GROUP, is also connected to GROUP-TYPE-ROLE through the ASSERTION attribute Value. A Group-Type-Role-Name might be "Sergeant" or "General". Note that these records can be reused across multiple ASSERTIONs, i.e., a Group-Type-Role-Name of "Sergeant" can be used for three different ASSERTIONs dealing with three different non-commissioned officers. In the children of a union example above, the Group-Type-Role-Name might be "3rd".

There are one to many GROUP-TYPE-ROLEs for each GROUP-TYPE. A Group-Type-Name in one case might be "U.S. Army", the type of group that had a "Sergeant" role. But there would be other GROUP-TYPE-ROLEs for the U.S. Army such as Lieutenant, Corporal, and so forth. Since many groups have rankings, there is a Sequence-Number to sort them properly.

The chart below shows possible values for selected attributes in the group entities.

ASSERTION	
Subject1-Type =	Persona
Subject1-ID =	Persona-ID
Subject2-Type =	Group
Subject2-ID =	Group-ID
Value =	Group-Type-Role-ID
GROUP	
Group-ID =	Match ASSERTION Subject
Group-Name =	(Children of union of) John Smith and Mary Jones
GROUP-TYPE	
Group-Type-Name =	Children of union
GROUP-TYPE-ROLE	
Group-Type-Role-ID =	Match ASSERTION Value
Group-Type-Role-Name =	Child
Sequence-Number =	3

There are several ***rules***, however, that govern the way the Lexicon group believes that the ASSERTION entity should handle GROUP statements.

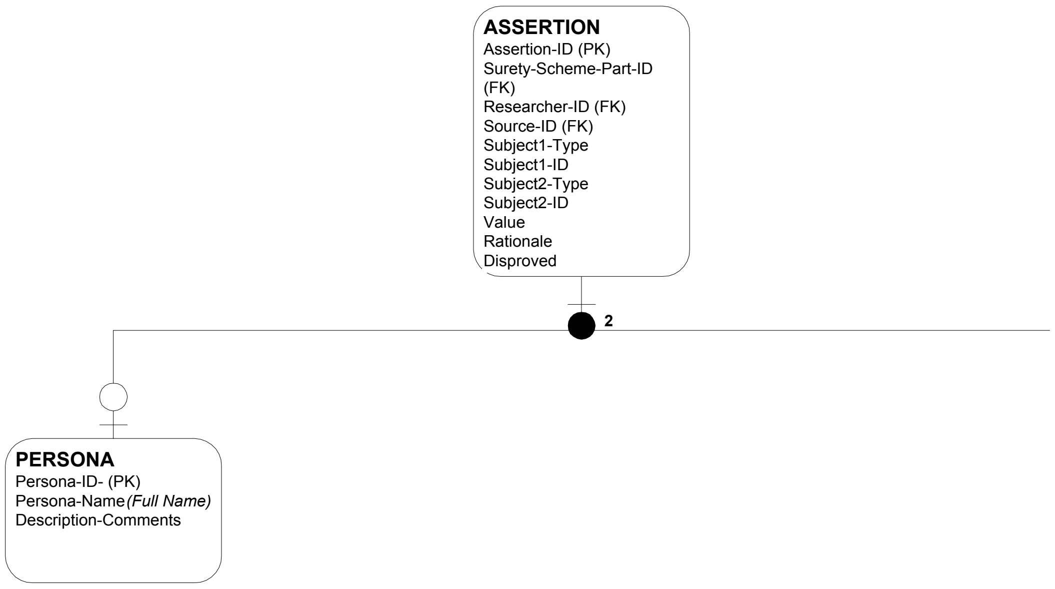
- A *new* GROUP cannot be Subject1. A GROUP must be formed by making a membership ASSERTION about a PERSONA, EVENT, CHARACTERISTIC, or an existing GROUP. However, if the researcher is making an ASSERTION about a GROUP of GROUPs, which is not only allowable but very useful, the existing GROUPs can each be (and must be!) Subject1; the new GROUP is Subject2.

- All members of a GROUP must be of the same type: PERSONA, EVENT, CHARACTERISTIC, or GROUP. Only GROUPs of the same atomic type can be further GROUPed.

- An instance created from a GROUP must be of the same type as the atomic members of the GROUP. For example, the researcher cannot create a new PERSONA from a GROUP of EVENTs.

See the following section for a similar restrictive rule about PERSONA.

5.4.6 Persona Entities



When one of the attachments, either subject 1 or subject 2, describes a person (but see the rules below for when a subject can be a PERSONA), it is connected to PERSONA. A Persona-Name might be "John Quincy Smith", but note that this is intended to be a composite name that the genealogist will use for an individual. The detailed name parts, based directly on evidence, are stored in CHARACTERISTIC-PART.

Recall from the discussion that the RESEARCHER makes an ASSERTION from evidence about a PERSONA. Raw evidence from a data gathering expedition is attached to many different PERSONAs, all of whom may share the same name, but at the stage of gathering evidence, the RESEARCHER is not sure that these are in fact the same person.

Eventually, the RESEARCHER may gather some of these individuals as a single PERSONA when there is sufficient evidence. In that case, there is an ASSERTION of course, and the result is that the group mechanism can be used to group multiple PERSONAs that the RESEARCHER feels are the same PERSONA. In that case, there will not be a sequence number for each if the concept is identity, that is, all the PERSONAs represent data for the same person. Alternatively, the sequence number can be used to indicate a change of names used over a lifetime, if that's appropriate.

Note that PERSONAs are created at two and only two times as follows.

- They are created directly from SOURCE fragments. A new PERSONA must be created whenever ASSERTIONs are created from SOURCE fragments.[8]

- They are created when making a new ASSERTION from two or more existing ASSERTIONs. This involves merging two or more existing PERSONAs into one new PERSONA using the GROUP concept.

These observations lead to the following ***rule***.

- An *existing* PERSONA cannot be Subject2, i.e., the only time a PERSONA can be Subject2 is when it is a new PERSONA being created from a GROUP. When a new PERSONA is being created from a SOURCE fragment, it is always Subject1 with Subject2 being a CHARATERISTIC, EVENT, or GROUP.

We need to look at this rule as it applies to the following four possible combinations, the only ones that can be made with PERSONA as Subject2.

Subject1	*Subject2*
GROUP	PERSONA
EVENT	PERSONA
CHARACTERISTIC	PERSONA
PERSONA	PERSONA

Thus, the common form of ASSERTION where PERSONA is Subject2 is where it is being created from a group of existing PERSONAs that the researcher wishes to bring together as follows.

GROUP173 (which includes PERSONAs 1 through 6, say) is PERSONA7.

Thus "GROUP is a new PERSONA" is allowable. Consider the following two ASSERTION forms, however, that do not make any sense.

EVENT1 is PERSONA2.
CHARACTERISTIC1 is PERSONA2.

Now consider some of the PERSONA to PERSONA ASSERTION forms. Although they make sense, the Lexicon group decided that it would be stronger to use the GROUP entity.

[8] We do recognize that in some cases we may have an atomic-level SOURCE statement like the following: "John Smith was born on 12 October 1855 with red hair." In this case, we do not require two separate ASSERTIONs, each pointing to a different John Smith as we normally do with even closely related statements. (We do, however, require two separate ASSERTIONs pointing to the *same* PERSONA; two pieces of data require two ASSERTIONs.) If these two pieces of data could *in any way* be interpreted as *possibly* applying to two different people, we would begin with separate ASSERTIONs pointing to two different PERSONAs and later, through diligent investigation and genealogical deduction, conclude that these two pieces of data do, in fact, apply to the same person and not to, say, a father and son of the same name. The preference, however, is to always err on the side of assuming that two people, even with the same name, are *not* the same person. This example is a relatively rare exception. A similar example, however, would include the EVENTs "Died" and "Buried" which might well come from the same SOURCE fragment. Similarly, a SOURCE fragment that states that a particular person "…aged 24 years, was married on <date>…" would allow us to make an ASSERTION about this PERSONA's birth event and marriage event; in this case we know it is the same PERSONA. Note, however, that these are exceptions. Often we must assume that the two people are *not* the same and we must make separate PERSONA statements until we can later group them together.

Here are some sample PERSONA to PERSONA forms that should be handled through GROUP.

PERSONA1 is the same as PERSONA2.
PERSONA1 is the father of PERSONA2.
PERSONA1 was born before PERSONA2.
PERSONA1 was taller than PERSONA2.

The purpose of the rule, however, is not to limit the *use* of PERSONA, but to limit the *creation* of PERSONA to the two circumstances listed. You can create a new PERSONA, as indicated above, by the following.

- Simply assert something about a PERSONA.

- Assert that a PERSONA is formed from a GROUP (of PERSONAs).

It's important to follow this reasoning a little further to understand why the Lexicon group limited PERSONA in this way. Although the S1 statement type that ties together two people can be used to express a relationship between two people (P1 / is the father of / P2) fairly directly, a piece of evidence that points to a father and 6 children requires 6 pairs of statements. If there were other relations mentioned at the same time, it becomes much more complex. Consider the simplest relationship form as expressed through the GROUP concept.

Subject1	*Subject2*	
PERSONA1	GROUP1	(with role "Father")
PERSONA2	GROUP1	(with role "Son")

This requires the same number of ASSERTION statements, and allows for more complex situations.

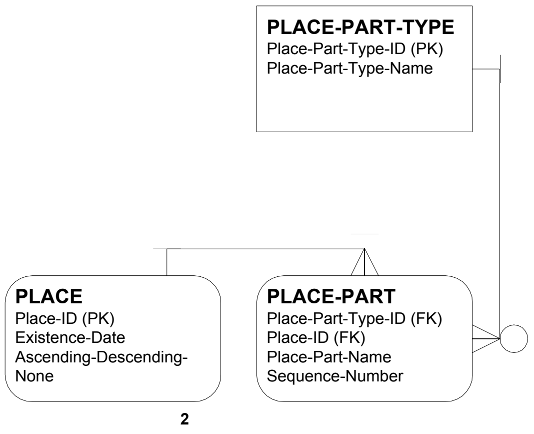
The process of aggregating individuals into higher level PERSONAs becomes the following.

Subject1	*Subject2*
PERSONA51	GROUP17
PERSONA56	GROUP17
GROUP17	PERSONA81

The important concept here is that PERSONA81 is created from merging PERSONA51 and PERSONA56, meaning that the researcher has determined that these are the same person. But if the researcher later discovers that this is not correct, they can be taken apart from the constructed PERSONA by tracing backwards and breaking the link (with the appropriate rationale so the researcher can remember a few years from now why he or she did this). We have not lost the original PERSONAs this way because they (and their evidence) still exist independently as PERSONA51 and PERSONA56.

See section 5.4.5 Group Entities on page 38 for a discussion of similar rules about GROUP.

5.4.7 Place Entities



The final entities in the Conclusional Submodel deal with place, and are connected extensively throughout the total model. All connections flow to PLACE which has an attribute indicating the date range of existence of the PLACE, so that both current and historical PLACEs can be stored. The attribute Ascending-Descending-None indicates how the PLACE-PARTs are stored. While PLACE might refer, through the connected place entities, to a particular place like “Silver Spring, Montgomery County, Maryland”, each of those parts are stored in PLACE-PART for which there are one to many for each instance of PLACE.

It is insufficient to simply store the PLACE-PARTs. For each part, we want to know what type of place it is, so for every Place-Part-Name in PLACE-PART like “Silver Spring”, there is one Place-Part-Type-Name in PLACE-PART-TYPE like “City”, “County”, “Street”, “Hospital” and so forth. Of course each Place-Part-Type-Name like “County” applies to zero to many Place-Part-Names.

Note that there is no provision in the model for establishing a hierarchy of PLACE-PART-TYPEs so that data exists that shows “County” as part of “State”, “State” as part of “Country”, and specifically “Prince George’s County” as part of “Maryland”. This was left as a logical model extension.

6.0 DATA DEFINITIONS

Note that although individual definitions below do not indicate which data is mandatory, all data in the one, one to many, or many side of a relationship is considered mandatory. However, since the novice researcher may choose to leap to conclusions without supplying the complete chain of evidence, the assumption is that any system that implements this data model will automatically provide linking data as needed. The linking data, such as Citation-Part-Value in CITATION-PART, might say "No data provided" or a similar phrase.

6.1 ACTIVITY

ACTIVITY	
Type:	Dependent. Requires RESEARCH-OBJECTIVE through RESEARCH-OBJECTIVE-ACTIVITY.
Definition:	Contains information about an activity such as a SEARCH or an ADMINISTRATIVE-TASK that must be, or was, accomplished. ACTIVITY allows the researcher to translate RESEARCH-OBJECTIVEs into specific action items. Note that ACTIVITY has two sub-entities: ADMINISTRATIVE-TASK and SEARCH. It contains the attributes that are common to both sub-entity.
Primary Key:	Activity-ID
Foreign Keys:	(None)
Relationships:	One ACTIVITY is the result of zero to many RESEARCH-OBJECTIVEs (through RESEARCH-OBJECTIVE-ACTIVITY). The zero condition addresses random or spontaneous activities that are not part of a pre-planned RESEARCH-OBJECTIVE.
	One RESEARCH-OBJECTIVE results in zero to many ACTIVITYs.
	One RESEARCHER undertakes zero to many ACTIVITYs.
	One ACTIVITY is performed by one RESEARCHER.
	One ACTIVITY is about either an ADMINSTRATIVE-TASK or a SEARCH.
	An ADMINSTRATIVE-TASK is a type of ACTIVITY.
	A SEARCH is a type of ACTIVITY.

ACTIVITY ATTRIBUTES	
Name	***Description***
Activity-ID	The unique key in ACTIVITY that identifies this ACTIVITY.
Researcher-ID	The unique key in RESEARCHER that identifies the person who either did or will do this ACTIVITY.
Scheduled-Date	The date that the researcher plans to conduct this ACTIVITY.
Completed-Date	The date that the researcher completed this ACTIVITY. If this is blank, then the ACTIVITY has not been completed.
TypeCode	This indicates whether the ACTIVITY is an ADMINISTRATIVE-TASK or a SEARCH.
Status	This describes the status of the ACTIVITY. Besides the obvious category of "Completed" which is redundant with Completed-Date having a value, other status indicators might be "waiting", "on hold", or some other value.
Description	A short description of the ACTIVITY.
Priority	A code indicating the priority the researcher sets on this activity.
Comments	Any comments that are required about this ACTIVITY.

6.2 ADMINISTRATIVE-TASK

ADMINISTRATIVE-TASK	
Type:	Dependent. Requires ACTIVITY because it is a sub-entity.
Definition:	A sub-entity of ACTIVITY that holds information related to various administrative chores other than conducting a genealogical SEARCH. Currently this is a rather bland entity, and mostly serves to indicate that an ACTIVITY is *not* a SEARCH and thus does not have the additional attributes required of a genealogical SEARCH.
Primary Key:	See Activity-ID.
Foreign Keys:	Activity-ID (both a primary key and a foreign key) because ADMINISTRATIVE-TASK is a subtype of ACTIVITY.
Relationships:	An ADMINISTRATIVE-TASK is a sub-entity of ACTIVITY.
	Each ACTIVITY has either an ADMINISTRATIVE-TASK or a SEARCH.

ADMINISTRATIVE-TASK ATTRIBUTES	
Name	***Description***
Activity-ID	The unique key in ACTIVITY that identifies this ADMINISTRATIVE-TASK.

6.3 ASSERTION

ASSERTION	
Type:	Dependent. Requires numerous other entities including RESEARCHER, the GROUP entities, PERSONA, the EVENT entities, CHARACTERISTIC, and SURETY.
Definition:	Contains the lowest level raw conclusional data in a special atomic form. This involves an interpretation by the researcher ranging from trivial to complex. This entity also contains higher level conclusional data from lower level assertions, so that all assertions can be tracked through layers of reasoning back to their original evidential statement forms. *Assertions should not be deleted*, but an attribute (Disproved) exists to nullify erroneous conclusions so that the erroneous reasoning can be preserved and marked as believed to be no longer valid. Everyone's work has value, even if it is later proved to be wrong. Since all assertions are tagged according to their origin, it is possible to store other's assertions as well and identify that data as such. While most assertions are tied to particular SOURCE excerpts (the Content attribute in REPRESENTATION) or previous assertions, an assertion can apply to an entire SOURCE.
Primary Key:	Assertion-ID
Foreign Keys:	Surety-Scheme-Part-ID (in SURETY-SCHEME-PART)
	Researcher-ID (in RESEARCHER)
	Source-ID (in SOURCE)
Relationships:	Each ASSERTION has data about zero to one PLACE.
	Each ASSERTION was written by one RESEARCHER.
	Each ASSSERTION is about two subjects, and each subject is one of the following: PERSONA, EVENT, CHARACTERISTIC, or GROUP.
	Some ASSERTIONs are related to either GROUP-TYPE-ROLE or EVENT-TYPE-ROLE (through the Value attribute).
	Each ASSERTION depends on zero to one SURETY-SCHEME-PARTs.
	Each ASSERTION is the direct output of no more than one SOURCE. Some ASSERTIONs are not the direct output of any SOURCE, but are the output of one to many other ASSERTIONs (through ASSERTION-ASSERTION); note that many lower level ASSERTIONs are coupled to one higher level ASSERTION by pairing one at a time through ASSERTION-ASSERTION.

ASSERTION ATTRIBUTES	
Name	***Description***
Assertion-ID	A unique code that identifies each assertion.
Surety-Scheme-Part-ID	A pointer that indicates how sure the researcher is of this particular assertion.
Researcher-ID	A pointer that identifies the researcher who made this assertion. The person asserting can be the researcher, or a compiler from which the researcher obtained data. In group projects, there may be many researchers.
Source-ID	A pointer to the source that gave rise to this assertion, if the assertion is the result of a direct source and not another assertion.
Subject1-Type	Can be either PERSONA, EVENT, CHARACTERISTIC, or GROUP.
Subject1-ID	A pointer to the appropriate PERSONA, EVENT,

	CHARACTERISTIC, or GROUP attribute of ID.
Subject2-Type	Can be either PERSONA, EVENT, CHARACTERISTIC, or GROUP.
Subject2-ID	A pointer to the appropriate PERSONA, EVENT, CHARACTERISTIC, or GROUP attribute of ID.
Value (Role)	If the statement is of the appropriate type, the value of the object in the statement. Example: (hair color) red; (occupation) teamster; (sex) female. In some instances, value can be thought of as “Role” such as “Groom” or “Witness”.
Rationale	Narrative that explains the researcher’s basis for the assertion. This can be curt for simple or trivial assertions, or very extensive if necessary for more complex assertions created from a variety of conflicting sources.
Disproved?	A yes/no indicator that the genealogist no longer believes the assertion to be true. “Yes” or “true” means it is no longer true.

6.4 ASSERTION-ASSERTION

ASSERTION-ASSERTION	
Type:	Dependent. Requires ASSERTION twice (where it feeds multiple ASSERTIONs into a new ASSERTION).
Definition:	An associative entity that links ASSERTION to itself so that multiple prior ASSERTIONs be brought together into a new ASSERTION. As an example, four ASSERTIONs based on individual SOURCEs can be brought together to resolve or document discrepancies about the date of a person's birth.
Primary Key:	None
Foreign Keys:	Assertion-ID-Low (in ASSERTION)
	Assertion-ID-High (in ASSERTION)
Relationships:	An ASSERTION-ASSERTION has one input ASSERTION.
	An ASSERTION-ASSERTION has one output ASSERTION.

ASSERTION-ASSERTION ATTRIBUTES	
Name	***Description***
Assertion-ID-Low	The unique key in ASSERTION for which this instance (i.e., a physical record in a table) serves as the input.
Assertion-ID-High	The unique key in ASSERTION for which this instance (i.e., a physical record in a table) serves as the output.
Sequence-Number	A value that keeps a series of input and output ASSERTIONs in order, so that for example, 4 lower level ASSSERTIONs can be brought together into a higher level ASSERTION with the order of the low level ASSERTIONs preserved.

6.5 CHARACTERISTIC

CHARACTERISTIC	
Type:	Dependent. Requires ASSERTION and CHARACTERISTIC-PART.
Definition:	A CHARACTERISTIC is any data that distinguishes one person from another, such as an occupation, hair color, religion, name, and so forth. Most CHARACTERISTIC data consists of a single part value, but some data can be more complex and require the sequencing of many parts such as a person's name.
Primary Key:	Characteristic-ID
Foreign Keys:	Place-ID (in PLACE)
Relationships:	One CHARACTERISTIC has one to many CHARACTERISTIC-PARTs.
	One CHARACTERISTIC-PART is part of only one CHARACTERISTIC.
	One CHARACTERISTIC is the subject of one ASSERTION.
	One ASSERTION describes zero to two CHARACTERISTICs.
	One CHARACTERISTIC happens in one PLACE.
	One PLACE can be the location of zero to many CHARACTERISTICs.

CHARACTERISTIC ATTRIBUTES	
Name	***Description***
Characteristic-ID	Unique identifier that indicates which characteristic this is.
Place-ID	Unique identifier in PLACE that indicates the place associated with this CHARACTERISTIC. Note that this is not a characteristic of a place (such as "nice view of the mountains"), but a place where a characteristic was noted, e.g., "Tuscon" is the place where John Smith was employed as a stagecoach driver, a type of occupation and thus a characteristic of John Smith.
Characteristic-Date	The date associated with the CHARACTERISTIC. This can be a point date (e.g., a specific day, week, month, or year) or it can be a date range.
Ascending-Descending-None	The sorting order of the attached CHARACTERISTIC-PARTs.

6.6 CHARACTERISTIC-PART

CHARACTERISTIC-PART	
Type:	Dependant. Requires CHARACTERISTIC and CHARACTERISTIC-PART-TYPE.
Definition:	Most CHARACTERISTICs have a single CHARACTERISTIC-PART. For example, the characteristic "Occupation" typically has a single value. But since the data model defines a person's name as another characteristic, and since name is made up of parts such as given name, surname, suffix, and so forth, this entity is required to collect the parts of a CHARACTERISTIC.
Primary Key:	Characteristic-Part-ID
Foreign Keys:	Characteristic-ID
	Characteristic-Part-Type-ID
Relationships:	One CHARACTERISTIC-PART is part of one CHARACTERISTIC.
	One CHARACTERISTIC has one to many CHARACTERISTIC-PARTs.
	One CHARACTERISTIC-PART is of one CHARACTERISTIC-PART-TYPE.
	One CHARACTERISTIC-PART-TYPE is seen in zero to many CHARACTERISTIC-PARTs. For example, the Characteristic-Part-Type-Name "Mononame" (in CHARACTERISTIC-PART-TYPE) is seen in the Characteristic-Part-Name "Sitting Bull", "Geronimo", and "Blue Duck" (in CHARACTERISTIC-PART).

CHARACTERISTIC-PART ATTRIBUTES	
Name	***Description***
Characteristic-Part-ID	Unique key that identifies a specific characteristic part.
Characteristic-ID	Unique key that identifies the characteristic.
Characteristic-Part-Type-ID	Unique key that identifies a specific characteristic part type.
Characteristic-Part-Name	The actual name of the characteristic part, such as "Stagecoach driver", "Red", or "Mary.
Sequence-Number	The number that keeps the characteristic parts sorted in correct order.

6.7 CHARACTERISTIC-PART-TYPE

CHARACTERISTIC-PART-TYPE	
Type:	Independent.
Definition:	In the case of most characteristics, this entity provides a list of the one and only part, such as "Occupation", "Hair Color", "Medical Condition", and so forth. In the case of personal names, however, this entity provides a list of all the name parts such as "Given Name", "Surname", "Mononame", "Prefix", and so forth.
Primary Key:	Characteristic-Part-Type-ID
Foreign Keys:	None
Relationships:	One CHARACTERISTIC-PART-TYPE is manifested as zero to many CHARACTERISTIC-PARTs.
	One CHARACTERISTIC-PART is of exactly one CHARACTERISTIC-PART-TYPE.

CHARACTERISTIC-PART-TYPE ATTRIBUTES	
Name	***Description***
Characteristic-Part-Type-ID	Unique key that identifies each member of the CHARACTERISTIC-PART-TYPE.
Characteristic-Part-Type-Name	The actual name of the CHARACTERISTIC-PART-TYPE, such as "Mononame", "Nickname", or "Occupation".

6.8 CITATION-PART

CITATION-PART	
Type:	Dependent. Requires CITATION-PART-TYPE and SOURCE.
Definition:	Provides a place to store the actual citation part for a particular SOURCE, such as author, title, and publication place (in Citation-Part-Value). There are a large number of CITATION-PART-TYPEs since there are a large number of types of genealogical records.
Primary Key:	None
Foreign Keys:	Source-ID (in SOURCE)
	Citation-Part-Type-ID (in CITATION-PART-TYPE)
Relationships:	One CITATION-PART-TYPE appears in zero to many CITATION-PARTs, e.g., there are a lot of citations for different authors.
	One CITATION-PART is of one and only one CITATION-PART-TYPE.
	One SOURCE can have zero to many CITATION-PARTs, e.g., a particular SOURCE might have an author, an editor, a compiler, a translator, a place of publication, and many other citation parts.
	One CITATION-PART refers to only one SOURCE. For example, "Baltimore" refers to the place of publication for a single SOURCE; if another SOURCE was also published in Baltimore, there would be another instance in CITATION-PART.

CITATION-PART ATTRIBUTES	
Name	***Description***
Source-ID	The unique key in SOURCE for which this is a citation part.
Citation-Part-Type-ID	The unique key in CITATION-PART-TYPE that identifies the type of citation part that this is, such as "Publication City", "Author", or "Title". Note that this is merely the ID and not the actual words.
Citation-Part-Value	The actual value of this citation part, such as "Baltimore", "Thomas Smith", or "Wills of Prince George's County, Maryland 1695-1710"

6.9 CITATION-PART-TYPE

CITATION-PART-TYPE	
Type:	Independent. Data about CITATION-PART-TYPE can be entered without regard to any other entity.
Definition:	Contains a list of citation parts, the names of the pieces of data found in citations of all types, such as author, editor, title, and place of publication. Note that this entity does not contain the actual citation values such as "Baltimore".
Primary Key:	Citation-Part-Type-ID
Foreign Keys:	None
Relationships:	One CITATION-PART-TYPE can be found in zero to many CITATION-PARTs.
	One CITATION-PART belongs to one and only one CITATION-PART-TYPE.

CITATION-PART-TYPE ATTRIBUTES	
Name	***Description***
Citation-Part-Type-ID	Unique identifier for this particular CITATION-PART.
Citation-Part-Type-Name	The actual name of the citation part, such as author, compiler, editor, transcriber, or place of publication. There are more than a hundred different citation parts.

6.10 EVENT

EVENT	
Type:	Dependent. Requires EVENT-TYPE.
Definition:	An EVENT is any type of happening such as a particular wedding.
Primary Key:	Event-ID
Foreign Keys:	Event-Type-ID (in EVENT-TYPE)
	Place-ID (in PLACE)
Relationships:	One EVENT is of an EVENT-TYPE.
	One EVENT-TYPE is manifested in zero to many EVENTs.
	One EVENT is the subject of one ASSERTION.
	One ASSERTION describes zero to two EVENTs.
	One EVENT happens in one PLACE.
	One PLACE can have zero to many EVENTs.

EVENT ATTRIBUTES	
Name	***Description***
Event-ID	Unique identifier that indicates which event this is.
Event-Type-ID	Unique identifier that indicates to which EVENT-TYPE this event belongs.
Place-ID	Unique identifier in PLACE that indicates the place associated with this EVENT. In short, where did this EVENT take place?
Event-Name	The name of the event, such as "Marriage of John Smith and Mary Jones".
Event-Date	The date associated with the event. This can be a point date (e.g., a specific day, week, month, or year) or it can be a date range.

6.11 EVENT-TYPE

EVENT-TYPE	
Type:	Independent. Does not require any other entities.
Definition:	Because many events (e.g., marriages) have quite similar structures, it's more efficient to define a type of event in a template structure than to keep defining individual events that are the same. The EVENT-TYPE contains the name of a standard event while the details about the usual roles played in such an event appear as individual instances of EVENT-TYPE-ROLE.
Primary Key:	Event-Type-ID
Foreign Keys:	None
Relationships:	One EVENT-TYPE is manifested as zero to many EVENTs.
	One EVENT is of one and only one EVENT-TYPE.
	One EVENT-TYPE has one to many EVENT-TYPE-ROLEs.
	One EVENT-TYPE-ROLE belongs to one and only one EVENT-TYPE.

EVENT-TYPE ATTRIBUTES	
Name	***Description***
Event-Type-ID	Unique key that identifies a specific event type.
Event-Type-Name	The name of this event type. An example might be "Marriage" or "Wedding", or "Battle".

6.12 EVENT-TYPE-ROLE

EVENT-TYPE-ROLE	
Type:	Dependent. Requires EVENT-TYPE.
Definition:	The individual roles of a defined event type, such as “Chaplain” for a role in a military unit.
Primary Key:	Event-Type-Role-ID
Foreign Keys:	Event-Type-ID (in EVENT-TYPE)
Relationships:	Each EVENT-TYPE-ROLE belongs to only one EVENT TYPE.
	An EVENT-TYPE can have zero to many EVENT-TYPE-ROLEs. The zero condition is for unity, where there is only one event type role in the event type, meaning everyone in the event participated in the same capacity, such as “Witness”.
	An EVENT-TYPE-ROLE can appear in zero to many ASSERTIONs in the Value attribute.
	One ASSERTION is about zero or one EVENT-TYPE-ROLEs.

EVENT-TYPE-ROLE ATTRIBUTES	
Name	***Description***
Event-Type-Role-ID	Unique key that identifies each member of the EVENT-TYPE.
Event-Type-ID	Unique key that identifies the EVENT-TYPE to which these members belong.
Event-Type-Role-Name	The value that distinguishes the different members of the event type, such as role (bride, groom, witness).

6.13 GROUP

GROUP	
Type:	Dependent. Requires GROUP-TYPE.
Definition:	In genealogical data, there are group members for which we can't identify query conditions to return the set. In other words, membership in a group such as "men who worked on the Davison Road in August, 1851" may be important genealogically, but no other attributes will sufficiently code for this. Thus, those members need to be tagged as explicit members of one or more groups. Groups are also used in this data model for concepts such as a group of children for a union of a man and woman.
Primary Key:	Group-ID
Foreign Keys:	Group-Type-ID (in GROUP-TYPE)
	Place-ID (in PLACE)
Relationships:	One GROUP is of a GROUP-TYPE.
	One GROUP-TYPE is manifested in zero to many GROUPs.
	One GROUP is the subject of one ASSERTION.
	One ASSERTION describes zero to two GROUPs.
	One GROUP was brought together in one PLACE.
	One PLACE can have zero to many GROUPs.

GROUP ATTRIBUTES	
Name	***Description***
Group-ID	Unique identifier that indicates which group this is.
Group-Type-ID	Unique identifier that indicates to which GROUP-TYPE this group belongs.
Place-ID	Unique identifier in PLACE that indicates the place associated with this GROUP. In the example of a group of neighbors, it would be the small area where they lived. In the case of the Titanic passengers and crew, it might be the city that they sailed from, or it might be the location in the ocean of the disaster as appropriate to the researcher's genealogical needs. Some groups may not be associated with a place.
Group-Name	The name of the group.
Group-Date	The date associated with the group. This can be a point date (e.g., a specific day, week, month, or year) or it can be a date range.
Group-Criteria	The criteria for admission to the group. For example, one group might be all the neighbors listed in a particular document, while a second group is a similar group of neighbors listed in a second document, or the same document at a different time.

6.14 GROUP-TYPE

GROUP-TYPE	
Type:	Independent. Does not require any other entities.
Definition:	Because many groups (e.g., military groups) have quite similar structures, it's more efficient to define a type of group in a template structure than to keep defining individual groups that are the same. The GROUP-TYPE contains the name and the ordering characteristics of a standard group while the details about the standard group appear as individual instances of GROUP-TYPE-ROLE.
Primary Key:	Group-Type-ID
Foreign Keys:	None
Relationships:	One GROUP-TYPE is manifested as zero to many GROUPs.
	One GROUP is of one and only one GROUP-TYPE.
	One GROUP-TYPE has one to many GROUP-TYPE-ROLEs.
	One GROUP-TYPE-ROLE belongs to one and only one GROUP-TYPE.

GROUP-TYPE ATTRIBUTES	
Name	***Description***
Group-Type-ID	Unique key that identifies a specific group type.
Group-Type-Name	The name of this group type. An example might be "U.S. Army grades and ranks, 1810-1830" (or whatever). Another group might be "Neighbors Occupying Contiguous Property"
Ascending-Descending-None	What is the ordering scheme of this group?

6.15 GROUP-TYPE-ROLE

GROUP-TYPE-ROLE	
Type:	Dependent. Requires GROUP-TYPE.
Definition:	The standard individual members of a defined group type, such as "Private, Corporal, Sergeant", "Bride, Groom, Witness", or "Miner, Pit Boss, Superintendent".
Primary Key:	Group-Type-Role-ID
Foreign Keys:	Group-Type-ID (in GROUP-TYPE)
Relationships:	Each GROUP-TYPE-ROLE belongs to only one GROUP TYPE.
	A GROUP-TYPE can have zero to many GROUP-TYPE-ROLEs. The zero condition is for unity, where there is only one group type role in the group type, meaning everyone in the group is of the same rank or type, such as a group of neighbors.
	A GROUP-TYPE-ROLE can appear in zero to many ASSERTIONs in the Value attribute.
	One ASSERTION is about zero or one GROUP-TYPE-ROLEs.

GROUP-TYPE-ROLE ATTRIBUTES	
Name	***Description***
Group-Type-Role-ID	Unique key that identifies each member of the GROUP-TYPE.
Group-Type-ID	Unique key that identifies the GROUP-TYPE to which these members belong.
Group-Type-Role-Name	The value that distinguishes the different members of the group type, such as role (bride, groom, witness) or rank (captain, major, colonel).
Sequence Number	The alphanumeric sequence number that causes the highest ranked group type member to be sorted high. For example, if the group consisted of (in this short example) "Colonel, General", Colonel might be assigned a sequence number of 2 and General a 1 to indicated that General ranks above Colonel. In the case of roles, sequence number may be irrelevant and may only serve to order the list for presentation so that "bride, groom, minister, witness, flower girl, ring bearer" appear in that order and not alphabetically.

6.16 PERSONA

PERSONA	
Type:	Dependent. Requires ASSERTIONs to support the data.
Definition:	Contains the core identification for each individual in genealogical data, and allows information about similarly named or identically named people to be brought together, after suitable analysis, in the same aggregate individual. Because real human beings leave data tracks through time as if they were disparate shadow personas, this entity allows the genealogical researcher to tie together data from different personas that he or she believes belong to the same real person. The mechanism for this, discussed in the text, is to make different PERSONAs part of the same GROUP.
Primary Key:	Persona-ID
Foreign Keys:	None
Relationships:	One PERSONA is based on one ASSERTION. However, note that an ASSERTION may link one PERSONA to a GROUP, and thus many separate PERSONAs can be brought together into a higher level constructed PERSONA.
	One ASSERTION can describe zero or one PERSONAs.

PERSONA ATTRIBUTES	
Name	***Description***
Persona-ID	Unique key identifying a single PERSONA.
Persona-Name	The entire name that this PERSONA is known by. This can be a special instance from a single record (from SOURCE and REPRESENTATION) like "John Q. Smith", or it can be a composite name built up from many separate instances, such as "John Quincy (Butch) Smith", that never actually appear in any record, but which reflects the name the way the RESEARCHER wishes to tag the individual.
Description-Comments	Any narrative necessary to distinguish this person.

6.17 PLACE

PLACE	
Type:	Dependent. Requires PLACE-PART-TYPE.
Definition:	Contains the core information about a PLACE, but does not include the subparts that make up the hierarchical name of a PLACE.
Primary Key:	Place-ID
Foreign Keys:	None
Relationships:	One PLACE has one to many PLACE-PARTs.
	One PLACE-PART belongs to one PLACE.
	PLACE also has numerous one to zero-to-many relationships with entities like ASSERTION, GROUP, EVENT, CHARACTERISTIC, RESEARCHER, and REPOSITORY.

PLACE ATTRIBUTES	
Name	***Description***
Place-ID	Unique key that identifies a place name.
Existence-Date	A point date or date range describing when this place was in existence.
Ascending-Descending-None	Describes the order of the PLACE-PARTs.

6.18 PLACE-PART

PLACE-PART	
Type:	Dependent. Requires PLACE-PART-TYPE.
Definition:	Contains information about a specific place, but in a way that the hierarchical relationship of that place to other places is preserved. One instance of PLACE-PART might be Maryland, while another is Virginia. Through association with PLACE-PART-TYPE we would know that both instances are called a "State". Note that a PLACE-PART like "Montgomery" is part of many different PLACEs. It is a county in several different states, and it is also a city in Alabama. But each of these Montgomerys would appear as a different instance in PLACE-PART and be attached to a different PLACE as would be expected.
Primary Key:	(None)
Foreign Keys:	Place-Part-Type-ID (in PLACE-PART-TYPE)
	Place-ID (in PLACE)
Relationships:	An example of one PLACE-PART-TYPE (such as "State") is found in zero to many actual PLACE-PARTs (such as "Colorado").
	One PLACE-PART is of one PLACE-PART-TYPE.
	One PLACE-PART appears in one PLACE.
	One PLACE is made up of one to many PLACE-PARTs.

PLACE-PART ATTRIBUTES	
Name	***Description***
Place-Part-Type-ID	Unique key that identifies the type of place part that this is, e.g., "State" or "Country" or "County", etc.
Place-ID	Unique key that identifies the PLACE of which this is a part.
Place-Part-Name	The actual name of this place part, such as "Prince George's".
Sequence-Number	The number that keeps the PLACE-PARTs in order, either ascending or descending (or in no order).

6.19 PLACE-PART-TYPE

PLACE-PART-TYPE	
Type:	Independent. Does not require any other entities.
Definition:	Contains information about various schemes of organizing place data in a hierarchical or other fashion. Parts might include “Country”, “State”, “Province”, “County”, and “City/Town/Village”.
Primary Key:	Place-Part-Type-ID
Foreign Keys:	(None)
Relationships:	One PLACE-PART-TYPE has zero to many PLACE-PARTs.

PLACE-PART-TYPE ATTRIBUTES	
Name	***Description***
Place-Part-Type-ID	Unique key that identifies the PLACE-PART-TYPE.
Place-Part-Type-Name	The name of this PLACE-PART-TYPE, such as “State”, “County”, “Country”, “Ocean”, or “Hospital”.

6.20 PROJECT

PROJECT	
Type:	Independent. Does not depend on any other entities.
Definition:	Information about the genealogical research project. One project might consist of all information about a person's ancestors, both on the researcher's father's side, and on the researcher's mother's side. Another project is all the ancestors on only one side of the researcher's family, such as the mother's side; this researcher might have another project for the father's side. Another project is a one-name study. Other types of genealogical projects include a study of the descendants of a particular person or couple, and the descendants of a particular *group* of people. Finally, a project can be undertaken for another person, in which case there is a client associated with the project. Note that client data is shown as an undefined attribute on the model, but would actually be a model extension for professional genealogists.
Primary Key:	Project-ID
Foreign Keys:	None
Relationships:	One PROJECT is worked on by one to many RESEARCHERs (through RESEARCHER-PROJECT).
	One PROJECT has zero to many RESEARCH-OBJECTIVEs.
	One PROJECT relies on zero or one SURETY-SCHEME.
	One SURETY-SCHEME can be used for zero to many PROJECTs.

PROJECT ATTRIBUTES	
Name	***Description***
Project-ID	Unique identifier for the particular project.
Name	The name of the project, such as "John F. Kennedy Ancestors", or "Mayflower Descendants".
Description	A text description of the project that provides additional information about the scope of the project, or any other necessary supporting information.
Client Data	If the project is undertaken for a client, the client name and address is included. An actual implementation for commercial genealogical purposes might have one or more separate entities for client information so that, for example, one client could commission one or more projects. Other information such as billing rates, expense logs, hour logs, and invoicing could be part of that system, but is not included in this basic genealogical data model.

6.21 REPOSITORY

REPOSITORY	
Type:	Independent. Requires no other entities.
Definition:	Contains information about the place where data is found. While this typically would be information about a library or archives, it can also be information about a private citizen who holds genealogical material of interest, such as a diary, family bible, and so forth. Data in this entity is sometimes part of a citation; it is required if the reader of the output of the data must know specifically where to find the data.
Primary Key:	Repository-ID
Foreign Keys:	Place-ID (in PLACE)
Relationships:	One REPOSITORY exists in one PLACE.
	One PLACE has zero to many REPOSITORYs.
	One REPOSITORY has zero to many SOURCEs (through REPOSITORY-SOURCE).
	One SOURCE is found in zero to many REPOSITORYs. (The zero condition is when we have data about a SOURCE but do not know where it can be found.)

REPOSITORY ATTRIBUTES	
Name	***Description***
Repository-ID	Unique key that identifies the specific REPOSITORY.
Place-ID	Unique key that identifies the specific PLACE that this REPOSITORY is located.
Name	The full name of the REPOSITORY. If it is an individual instead of an institution, substitute the individual's data throughout.
Address	The address of the REPOSITORY.
Phone	The phone number of the REPOSITORY.
Hours	The hours that the REPOSITORY is open to the public.
Comments	Any pertinent comments about the repository, such as the need to obtain a researcher's card, restrictions on the use of laptops, etc.

6.22 REPOSITORY-SOURCE

REPOSITORY-SOURCE	
Type:	Dependent. Requires SEARCH, REPOSITORY, and SOURCE.
Definition:	An associative entity that ties together REPOSITORY and SOURCE in a many to many relationship. Each instance in this entity represents a particular SOURCE in a specific REPOSITORY.
Primary Key:	None
Foreign Keys:	Repository-ID (in REPOSITORY)
	Source-ID (in SOURCE)
	Activity-ID (in SEARCH)
Relationships:	One REPOSITORY-SOURCE describes either one SOURCE or one REPOSITORY or one of each.
	One SEARCH is conducted in zero to one REPOSITORYs (through REPOSITORY-SOURCE).
	One REPOSITORY is the scene of zero to many SEARCHs (through REPOSITORY-SOURCE).
	One SEARCH is conducted in zero to one SOURCEs (through REPOSITORY-SOURCE).
	One SOURCE provides data for zero to many SEARCHs (through REPOSITORY-SOURCE).
	One SOURCE is found in zero to many REPOSITORYs (through REPOSITORY-SOURCE).
	One REPOSITORY has zero to many SOURCEs (through REPOSITORY-SOURCE) that can be searched.

REPOSITORY-SOURCE ATTRIBUTES	
Name	***Description***
Repository-ID	Unique key that identifies a specific REPOSITORY.
Source-ID	Unique key that identifies a specific SOURCE.
Activity-ID	Unique key that identifies a specific SEARCH.
Call Number	The unique call number for a particular SOURCE in a particular REPOSITORY. Some REPOSITORYs use the same call number for the same SOURCE such as a federal censuses, but most materials have different call numbers. In some cases, there are multiple copies of a SOURCE in a REPOSITORY, and the researcher may wish to record which copy was the object of the SEARCH, particularly if the copy was not in good condition, and thus if the researcher wishes to SEARCH another copy.
Description	Any pertinent notes about the particular SOURCE in the REPOSITORY, such as notes describing the condition of the copy represented by the particular call number.

6.23 REPRESENTATION

REPRESENTATION	
Type:	Dependent. Requires SOURCE and REPRESENTATION-TYPE.
Definition:	Contains the representation of a SOURCE in a variety of multimedia formats as needed, including old fashioned text, plus it contains a pointer to a physical file if the representation cannot be stored within the data model.
Primary Key:	None
Foreign Keys:	Source-ID (in SOURCE)
	Representation-Type-ID (in REPRESENTATION-TYPE)
Relationships:	One REPRESENTATION is a manifestation of one SOURCE.
	One SOURCE has zero to many REPRESENTATIONs. The zero condition is useful for a SOURCE in which the researcher found nothing. The SEARCH in the SOURCE was significant and was recorded, but there is no REPRESENTATION, i.e., no photocopy, no text extract, no photo.
	One REPRESENTATION is of one REPRESENTATION-TYPE.
	One REPRESENTATION-TYPE is manifested in zero to many REPRESENTATIONs.

REPRESENTATION ATTRIBUTES	
Name	***Description***
Source-ID	Unique key that identifies the specific SOURCE.
Representation-Type-ID	Unique key that identifies the type of representation, such as text, TIF bitmap, or other type.
Physical-File-Code	If the REPRESENTATION is external to the data model, such as a stored photograph that is not scanned into a computer system, this code tells the researcher where the REPRESENTATION is physically filed or stored.
Medium	Often the SOURCE medium is paper, but it can be electronic, stone in the case of a tombstone, or other exotic media.
Content	The actual content of the REPRESENTATION. This can be text in the case of an abstract, extract, or transcription, or it can be other REPRESENTATIONs that can be stored within the confines of the actual implementation of the logical data model such as a bitmap that is stored in a computer application, or a sound file. If the content cannot be stored in the model, this is empty. An example would be a physical artifact like a souvenir glass from the World's Fair with the bride and groom's name and the marriage date; clearly we cannot store this electronically, but we could store a photograph of it electronically.
Comments	Any comments that are required to describe this REPRESENTATION.

6.24 REPRESENTATION-TYPE

REPRESENTATION-TYPE	
Type:	Independent.
Definition:	Contains a list of the types of representations of evidence, such as text, a TIF bitmap, a GIF bitmap, a WAV file, or other forms.
Primary Key:	Representation-Type-ID
Foreign Keys:	None.
Relationships:	One REPRESENTATION-TYPE describes zero to many REPRESENTATIONs.
	One REPRESENTATION is of one REPRESENTATION-TYPE.

REPRESENTATION-TYPE ATTRIBUTES	
Name	***Description***
Representation-Type-ID	Unique key that identifies the specific REPRESENTATION-TYPE.
Representation-Type-Name	The name of the REPRESENTATION-TYPE such as “Text”, “PCX Bitmap”, and so forth.

6.25 RESEARCH-OBJECTIVE

RESEARCH-OBJECTIVE	
Type:	Dependent. Requires PROJECT.
Definition:	Contains information about the RESEARCH-OBJECTIVEs that the RESEARCHER has determined are appropriate for the specific PROJECT. For example, one objective might be to "Find the father of John Smith."
Primary Key:	Research-Objective-ID
Foreign Keys:	Project-ID (in PROJECT)
Relationships:	One PROJECT has zero to many RESEARCH-OBJECTIVEs.
	One RESEARCH-OBJECTIVE applies to one PROJECT.
	One RESEARCH-OBJECTIVE results in zero to many ACTIVITYs (through RESEARCH-OBJECTIVE-ACTIVITY).
	One ACTIVITY is associated with zero to many RESEARCH-OBJECTIVEs (through RESEARCH-OBJECTIVE-ACTIVITY).

RESEARCH-OBJECTIVE ATTRIBUTES	
Name	***Description***
Research-Objective-ID	Unique key that identifies the specific RESEARCH-OBJECTIVE.
Project-ID	Unique key that identifies the PROJECT that this RESEARCH-OBJECTIVE belongs to.
Name	The name of the RESEARCH-OBJECTIVE.
Description	A more detailed description of the RESEARCH-OBJECTIVE.
Sequence-Number	A value that keeps the RESEARCH-OBJECTIVEs sorted in any order that the RESEARCHER wants.
Priority	The priority assigned to this RESEARCH-OBJECTIVE by the RESEARCHER.
Status	The status of this RESEARCH-OBJECTIVE such as "Open" or "Closed".

6.26 RESEARCH-OBJECTIVE-ACTIVITY

RESEARCH-OBJECTIVE-ACTIVITY	
Type:	Dependent. Requires RESEARCH-OBJECTIVE and ACTIVITY.
Definition:	Associative entity that breaks up the many to many relationship between RESEARCH-OBJECTIVE and ACTIVITY.
Primary Key:	None
Foreign Keys:	Research-Objective-ID (in RESEARCH-OBJECTIVE)
	Activity-ID (in ACTIVITY)
Relationships:	One RESEARCH-OBJECTIVE has zero to many RESEARCH-OBJECTIVE-ACTIVITYs.
	One RESEARCH-OBJECTIVE-ACTIVITY supports one RESEARCH-OBJECTIVE.
	One RESEARCH-OBJECTIVE-ACTIVITY results in one ACTIVITY.
	One ACTIVITY is associated with zero to many RESEARCH-OBJECTIVE-ACTIVITYs.

RESEARCH-OBJECTIVE-ACTIVITY ATTRIBUTES	
Name	***Description***
Research-Objective-ID	Unique key that identifies the specific RESEARCH-OBJECTIVE.
Activity-ID	Unique key that identifies the ACTIVITY that this RESEARCH-OBJECTIVE-ACTIVITY supports.

6.27 RESEARCHER

RESEARCHER	
Type:	Independent. Data about RESEARCHER can be entered without regard to any other entity.
Definition:	Information about a genealogical researcher that identifies who is responsible for any particular piece of data in the system.
Primary Key:	Researcher-ID
Foreign Keys:	None
Relationships:	One RESEARCHER participates in zero to many PROJECTs (through RESEARCHER-PROJECT).
	The data for one PROJECT comes from one to many RESEARCHERs (through RESEARCHER-PROJECT).
	A RESEARCHER lives in one PLACE.
	A RESEARCHER performs zero to many SEARCHs.
	A SEARCH is made by one and only one RESEARCHER.
	A RESEARCHER makes zero to many ASSERTIONs.
	An ASSERTION is made by one and only one RESEARCHER.

RESEARCHER ATTRIBUTES	
Name	***Description***
Researcher-ID	Unique identifier for this particular RESEARCHER.
Name	The full name of the researcher, suitable for reports.
Address	The address of the researcher. Part of the address is connected to Place-ID in PLACE.
Comments	Comments about the researcher, if necessary.

6.28 RESEARCHER-PROJECT

RESEARCHER-PROJECT	
Type:	Dependent. Requires RESEARCHER and PROJECT.
Definition:	Associative entity that ties together RESEARCHER and PROJECT so that one RESEARCHER can work on zero to many PROJECTs and one PROJECT can have one to many RESEARCHERs.
Primary Key:	None
Foreign Keys:	Researcher-ID (in RESEARCHER)
	Project-ID (in PROJECT)
Relationships:	One RESEARCHER-PROJECT describes one PROJECT.
	One PROJECT has one to many RESEARCHER-PROJECTs.
	One RESEARCHER-PROJECT is worked on by one RESEARCHER.
	One RESEARCHER works on zero to many RESEARCHER-PROJECTs.

RESEARCHER-PROJECT ATTRIBUTES	
Name	***Description***
Researcher-ID	Unique key that indicates which RESEARCHER.
Project-ID	Unique key that indicates which PROJECT.
Role	If it is necessary to describe the role that a particular RESEARCHER played, this field can differentiate different people on a group effort.

6.29 SEARCH

SEARCH	
Type:	Dependent. Requires SOURCE or REPOSITORY. Also requires RESEARCHER. Also a subtype of ACTIVITY (along with ADMINISTRATION-TASK) and thus usually but not always requires a RESEARCH-OBJECTIVE as well.
Definition:	A specific examination of a SOURCE to find information, usually based on a RESEARCH-OBJECTIVE, although a SEARCH can be conducted with no RESEARCH-OBJECTIVE in mind, particularly where it is a casual search based on an unplanned or unexpected opportunity. The concept of SEARCH is heavily influenced by the need to record what data the RESEARCHER looked for in a particular SOURCE on a particular research trip, to avoid having to look up that data again. A SEARCH can return specific data, or a SEARCH can result in not finding the data searched for, which is, of course, significant in itself.
Primary Key:	See Activity-ID (in ACTIVITY)
Foreign Keys:	Source-ID (in SOURCE)
	Repository-ID (in REPOSITORY)
Relationships:	One SOURCE takes place in one REPOSITORY-SOURCE.
	One SEARCH is conducted in zero to one REPOSITORYs (through REPOSITORY-SOURCE).
	One REPOSITORY is the scene of zero to many SEARCHs (through REPOSITORY-SOURCE).
	One SEARCH is conducted in zero to one SOURCEs (through REPOSITORY-SOURCE).
	One SOURCE provides data for zero to many SEARCHs (through REPOSITORY-SOURCE).

SEARCH ATTRIBUTES	
Name	***Description***
Activity-ID	Unique key that identifies a SEARCH as a subtype of an ACTIVITY.
Source-ID	Unique key that identifies a SOURCE that this SEARCH took place in. If the SEARCH was a general SEARCH in a REPOSITORY, for example to determine what suitable materials the REPOSITORY contains, this may be blank.
Repository-ID	Unique key that identifies a REPOSITORY. This is a required attribute and cannot be blank.
Searched-For	The text, such as a surname and certain variations, searched for.

6.30 SOURCE

SOURCE	
Type:	Independent. Does not require any other entities.
Definition:	A collection of data useful for genealogical research such as a will book, a deed book, a compiled genealogy in book pr periodical form, an electronic database, or similar collection. SOURCEs include both primary and secondary works. Generally a SOURCE will have one or more documents such as specific wills inside the will book; in many cases there will be additional levels of SOURCE. In some cases, the SOURCE has only one level; what we might think of as a document is conceptually the same as the SOURCE. Thus, SOURCE is self-referential and can handle data of any reasonable number of hierarchical levels.
Primary Key:	Source-ID
Foreign Keys:	Higher-Source-ID (in SOURCE)
	Subject-Place-ID (Place-ID in PLACE)
	Jurisdiction-Place-ID (Place-ID in PLACE)
	Researcher-ID (in RESEARCHER)
Relationships:	One high level SOURCE has zero to many lower level SOURCEs.
	One low level SOURCE can belong to zero to one higher level SOURCE.
	One SOURCE is part of zero to many SOURCE-GROUPs (through SOURCE-GROUP-SOURCE).
	One SOURCE-GROUP contains zero to many SOURCEs (through SOURCE-GROUP-SOURCE).
	One SOURCE is the object of zero to many SEARCHs (through REPOSITORY-SOURCE).
	One SEARCH takes place in zero to one SOURCEs (through REPOSITORY-SOURCE). A SEARCH either takes place in a RESPOSITORY (general SEARCH) or it takes place in a SOURCE (specific SEARCH).
	One SOURCE is found in zero to many REPOSITORYs (through REPOSITORY-SOURCE). The zero condition indicates a SOURCE that cannot presently be associated with a particular REPOSITORY, i.e., the RESEARCHER knows it exists, but does not know where to find it.
	One REPOSITORY contains zero to many SOURCEs (through REPOSITORY-SOURCE).
	One SOURCE was originally compiled about one jurisdiction PLACE and one SOURCE is about a person from one PLACE (which may not be the same as the jurisdiction PLACE). (Thus one SOURCE is about exactly two PLACEs.)
	One PLACE is associated with zero to many SOURCE jurisdictions and zero to many SOURCE persons.
	One SOURCE has zero to many REPRESENTATIONs, meaning that we might have text representing the SOURCE as well as a photocopy or a photograph, or some other multimedia REPRESENTATION.
	A REPRESENTATION applies to only one SOURCE. (In an example like a photocopy that contains two small wills, the RESEARCHER can simply list the same Physical-File-Code for both REPRESENTATIONs.)
	One SOURCE has many CITATION-PARTs. For example, at the book level, a SOURCE has a title, author, place of publication, and many other parts.
	One CITATION-PART cites one SOURCE.

SOURCE ATTRIBUTES	
Name	***Description***
Source-ID	Unique key that identifies the SOURCE.
Higher-Source-ID	Unique key that identifies the next higher level SOURCE associated with this SOURCE.
Subject-Place-ID	Unique key that identifies the PLACE of the subject of this SOURCE. Example: A record in North Carolina describes a person and their activities in Georgia. Georgia is the subject place, and North Carolina is the record jurisdiction place.
Jurisdiction-Place-ID	Unique key that identifies the PLACE of the jurisdiction of the record.
Researcher-ID	Unique key in RESEARCHER that identifies the person who gathered this SOURCE record.
Subject-Date	The date associated with the subject of this SOURCE. Note that there can be a somewhat different date associated with each level of a multi level SOURCE, such as a date range for a will book, and a more specific date for the will itself, and then perhaps other dates associated with small pieces of information in the will.
Comments	Any comments about the SOURCE that are required. If the SOURCE is at the level of a whole "book" for example, such as a will book, the comments may describe the poor condition and the difficulty in reading most entries.

6.31 SOURCE-GROUP

SOURCE-GROUP	
Type:	Independent. Does not require any other entities.
Definition:	Contains a list of groups of SOURCEs such as “Federal Census”, “Will”, “Deed”, and so forth. This is necessary in some cases so that data can be searched, selected, sorted, and grouped by type of SOURCE; without an explicit SOURCE-GROUP it may not be clear what type of record is represented by the SOURCE, although in most cases the title is explicit enough. However, some researchers may wish to group SOURCEs of particular interest such as “New England Sources”, “Massachusetts Sources”, or “Boston Sources”. Consequently, a SOURCE can be in more than one group and in the examples above more than one group scheme.
Primary Key:	Source-Group-ID
Foreign Keys:	None
Relationships:	One SOURCE-GROUP represents the type of source for zero to many SOURCEs (through SOURCE-SOURCE-GROUP).
	One SOURCE belongs to zero to many SOURCE-GROUPs (through SOURCE-SOURCE-GROUP). (Although zero to one is the normal condition, this allows the RESEARCHER to use multiple grouping concepts for the same SOURCEs.)

SOURCE-GROUP ATTRIBUTES	
Name	***Description***
Source-Group-ID	Unique key that identifies the SOURCE-GROUP.
Source-Group-Name	The name of the SOURCE-GROUP such as “Will”, “Deed”, or “Tombstone”.

6.32 SOURCE-GROUP-SOURCE

SOURCE-GROUP-SOURCE	
Type:	Dependent. Requires SOURCE-GROUP and SOURCE.
Definition:	An associative entity that ties together SOURCE and SOURCE-GROUP in a many to many relationship.
Primary Key:	None
Foreign Keys:	Source-ID (in SOURCE)
	Source-Group-ID (in SOURCE-GROUP)
Relationships:	One SOURCE-GROUP has zero to many SOURCEs (through SOURCE-SOURCE-GROUP).
	One SOURCE belongs to zero to many SOURCE-GROUPs (through SOURCE-SOURCE-GROUP).

SOURCE-GROUP-SOURCE ATTRIBUTES	
Name	***Description***
Source-ID	Unique key that identifies the SOURCE.
Source-Group-ID	Unique key that identifies the SOURCE-GROUP.

6.33 SURETY-SCHEME

SURETY-SCHEME	
Type:	Independent. Requires no other entities.
Definition:	The scheme used to establish the surety level of assertions made for a particular project. Different researchers may use different schemes such as "1 to 3" or "1 to 5" or "E, F, P", and in order to understand the researcher's evaluations, it is necessary to understand the particular scheme in use. Our standard requires that the scheme sort high as the most reliable.
Primary Key:	Surety-Scheme-ID
Foreign Keys:	None
Relationships:	One SURETY-SCHEME is used in zero to many PROJECTs.
	One PROJECT uses zero to one SURETY-SCHEMEs.
	One SURETY-SCHEME has one to many SURETY-SCHEME-PARTs.
	One SURETY-SCHEME-PART belongs to one SURETY-SCHEME.

SURETY-SCHEME ATTRIBUTES	
Name	***Description***
Surety-Scheme-ID	Unique identifier for each surety scheme.
Surety-Scheme-Name	The name of the surety scheme.
Surety-Scheme-Description	A general description of the SURETY-SCHEME, if necessary.

6.34 SURETY-SCHEME-PART

SURETY-SCHEME-PART	
Type:	Dependent. Requires SURETY-SCHEME.
Definition:	Contains information about the individual parts of the SURETY-SCHEME. For example, if the scheme is simply 1 to 5, then this entity lists the 5 levels and no further information may be required, other than a sequencer to determine what order the parts are in. However, the RESEARCHER may (and really should) choose to more fully explain what each surety level means.
Primary Key:	Surety-Scheme-Part-ID
Foreign Keys:	Surety-Scheme-ID
Relationships:	One SURETY-SCHEME-PART is part of one SURETY-SCHEME.
	One SURETY-SCHEME has one to many SURETY-SCHEME-PARTs. The one condition is unusual, but is useful if a researcher chooses to treat all evidence as the same surety level.
	One SURETY-SCHEME-PART describes zero to many ASSERTIONs.
	One ASSERTION is categorized by zero to one SURETY-SCHEME-PARTs. Note that in this model the RESEARCHER assigns surety levels to the ASSERTIONs made from the direct evidence, not to the evidence itself, but functionally this amounts to the same thing since the ASSERTIONs are closely coupled to the various levels of SOURCE.

SURETY-SCHEME-PART ATTRIBUTES	
Name	***Description***
Surety-Scheme-Part-ID	Unique identifier that indicates which SURETY-SCHEME-PART this is.
Surety-Scheme-ID	Unique identifier that determines to what SURETY-SCHEME this part belongs.
Surety-Scheme-Part-Name	The name of the SURETY-SCHEME-PART such as "1" or "G".
Surety-Scheme-Part-Description	An explanation of what the SURETY-SCHEME-PART means. If "2" in one scheme or "G" in another stands for "Good", for example, how does the RESEARCHER define "good"? What kinds of data would routinely be assigned a level of "good" instead of some other category?
Sequence-Number	An alphanumeric sequencer that sorts the most reliable SURETY-SCHEME-PART high. For example, if "1, 2, 3" is used by one researcher, and "1" is the most sure, then the corresponding ranks might also be 1, 2, and 3. If another researcher uses the same "1, 2, 3" but 3 is the most sure, then the corresponding ranks might be C, B, and A to force the list to come out 3, 2, 1.

APPENDIX A: PRINCIPLES OF LOGICAL DATA MODELING

This section is intended to introduce non-technical readers to the concepts of data modeling, and for readers who already understand data modeling methodology, this section will provide a quick overview of the conventions used in this particular model.

A.1 DATA MODELING AND THE RELATIONAL MODEL

There are three primary methods of storing data in a modern computer Database Management System (DBMS): relational, hierarchical, and network. Hierarchical data is organized using nested data structures as the name implies. Network data is generalized and stored in a network of nodes and links. Relational data is stored in a tabular format.

Almost all business data that resides on microcomputers is stored using a relational DBMS (RDBMS)[9], and this has proved to be a useful way to look at most kinds of data. It should be noted, however, that not all data fits nicely into the relational model, and in fact some of the basic genealogical data, such as that used to produce either an ancestor chart or a descendant chart is clearly not tabular at all, but is obviously hierarchical in nature. So, while any of the three kinds of DBMS structures could be used for any kind of data, some data fits one model better than another.

Because most genealogical data will eventually go into an RDBMS, and because data modeling most closely follows the relational model, we will take a quick look at relational database definitions. Many readers will already have a familiarity with the basic terms of physical databases, and this understanding will transfer easily to data modeling concepts. Note, however, that data modeling produces a logical model that is *independent* of the way the data actually gets implemented. So, while it may be useful to compare RDBMS terminology and data modeling terminology, we will leave the actual specification of physical tables to the developers.

The first concept is a **table**, a place to store similar data. A table was often called a file when the data was actually placed in a separate physical location on a disk, and some people tend to think of tables as files. An example of a table is one that contains the titles of books such as the following.

[9] There is considerable debate among systems engineers about the degree to which any particular DBMS such as dBASE, Oracle, Sybase, or Access actually fits the relational model as precisely defined many years ago in a classic work by Date and Codd. For our purposes, however, we don't care whether a particular software package is "truly" relational, because we will be working at an abstract level *above* the physical packages, defining how genealogical data is related. How it actually gets implemented is the developer's concern, and not ours.

TITLE

Title-ID	Title-Name	Copyright
122	Cite Your Sources	1980
131	Concise Genealogical Dictionary	1989
153	Encyclopedia of American Family Names	1995
174	Pitfalls in Genealogical Research	1987

Note that the data has been organized into rows and columns. We call a row a **record**, so that the first record in this table has all the data that refers to the book *Cite Your Sources*.

Looking at the data differently, we notice that each column has a name, and we call this a **field**. So, the Copyright field, for example, contains the copyright year for each record. The Title-Name field obviously contains the title of each book, and the use of the Title-ID field, which is a unique number assigned to each record, will become clear shortly.

If we were building a physical database, the TITLE table would contain the limited amount of data that we're interested in about each title (in this simple example), except for the author. (A real system, of course would have additional data such as publisher, ISBN, and so forth.) If we realized that one author might have written more than one book, we might have designed the system so that a single author entry would be **related** to several titles. This would have the advantage of not entering the same author over and over.

Consider the following *bad* example.

BAD-TITLE

Title-ID	Title-Name	Copyright	Author-Name
117	Cardinal of the Kremlin, The	1988	Clancy, Tom
122	Cite Your Sources	1980	Lackey, Richard
125	Clear and Present Danger	1989	Clancy, Tom
131	Concise Genealogical Dictionary	1989	Harris, Glen & Maurine
148	Debt of Honor	1994	Clancy, Tom
153	Encyclopedia of Am. Fam. Names	1995	Chesler & Robb
155	Firm, The	1991	Grisham, John
159	Foucault's Pendulum	1988	Eco, Umberto
163	Island of the Day Before, The	1994	Eco, Umberto
167	Name of the Rose, The	1980	Eco, Umberto
169	Pelican Brief, The	1992	Grisham, John
174	Pitfalls in Genealogical Research	1987	Rubincam, Milton
175	Researcher's Guide to Am. Gen.	1973	Greenwood, Val
181	Time to Kill, A	1989	Grisham, John
189	Without Remorse	1993	Clancy, Tom

Without discussing the technical details (right now[10]) of putting data in what are called normal forms, e.g., first normal form, second normal form, and so forth, we can see that the BAD-TITLE table doesn't do a very good job of storing the data. Although it doesn't look like much waste, we have three records for Umberto Eco, and multiple records for other authors as well. If we had additional author information like date of birth, sex, literary agent's name, and so forth, storing this same information repeatedly would clearly be undesirable.

[10] See section A.2 THE RULES OF NORMALIZATION on page 84, for a more complete explanation of normalization.

There's another problem with this table in that some titles, such as *Concise Genealogical Dictionary*, have more than one author. Depending on our application, we might be willing to treat a particular occurrence of a pair of authors to be the same as a single new author, but in a large system with many permutations of authors teaming up, as for example in writing academic papers, this would become unwieldy. Clearly, we need to establish a separate table for the authors. Consider the following table.

AUTHOR

Author-ID	**Last-Name**	**First-Name**	**Sex**	**Other-Stuff**
1	Chesler	Andrew	M	…
2	Clancy	Tom	M	
3	Eco	Umberto	M	
4	Grisham	John	M	
5	Harris	Glen	M	
6	Harris	Maurine	F	
7	Lackey	Richard	M	
8	Robb	H. Amanda	F	
9	Rubincam	Milton	M	

We could simply take BAD-TITLE and substitute an Author-ID column for the Author-Name column. This would make the redundant entries much shorter, particularly if there was a lot of additional author information, but we still haven't solved the problem of one author writing multiple titles, and one title having multiple authors.

The solution is to remove the Author-Name column completely from the BAD-TITLE table, and create a new table called AUTHOR-TITLE to hold just the ID information from both tables as shown on the next page.

AUTHOR-TITLE

Author-ID	Title-ID
1	153
2	117
2	125
2	148
2	189
3	159
3	163
3	167
4	155
4	169
5	131
4	181
6	131
7	122
8	153
9	174

TITLE

Title-ID	Title-Name	Copyright
117	Cardinal of the Kremlin, The	1988
122	Cite Your Sources	1980
125	Clear and Present Danger	1989
131	Concise Genealogical Dictionary	1989
148	Debt of Honor	1994
153	Encyclopedia of American Family Names	1995
155	Firm, The	1991
159	Foucault's Pendulum	1988
163	Island of the Day Before, The	1994
167	Name of the Rose, The	1980
169	Pelican Brief, The	1992
174	Pitfalls in Genealogical Research	1987
175	Researcher's Guide to American Genealogy	1973
181	Time to Kill, A	1989
189	Without Remorse	1993

Although the AUTHOR-TITLE table may seem awkward to process as a human being, it's very efficient for a computer. Note that it solves the problem of multiple authors for a single title; book 153 in AUTHOR-TITLE is associated with author 1 and author 8, while book 131 is associated with authors 5 and 6. Similarly, author 2 is associated with 4 books, and so forth.

A.2 THE RULES OF NORMALIZATION

The rules of normalization are unfortunately much easier to grasp with examples than they are to understand from the rules themselves. However, since it's likely that comments on the normalization of data in the data model[11] will only come from readers who are sufficiently immersed in data modeling methodology, no tutorial on the subject (complete with extensive examples) will be presented here. Instead, the rules will be noted below with a brief explanation of what they mean, and we will state that the data model is normalized to the Fourth Normal Form. The statements are in terms of relational tables as we discussed above, but note that the same rules apply to the data model itself, and that the appropriate data modeling terminology can be substituted.

A.2.1 First Normal Form: Eliminate Repeating Groups

A table should have only those fields that are logically grouped together. Separate any fields that repeat, and put them in their own table. Give each table a primary key.

The BAD-TITLE table was an example of a table that had two kinds of information mixed together, title and author information, and the author information repeated (more than one author for one title[12]). This table violated First Normal Form, so we removed the author information to its own table.

A.2.2 Second Normal Form: Eliminate Redundant Data

If a field depends on only part of a multivalued key, it must be removed to a separate table. If we had included a publisher field in BAD-TITLE, we would have seen the same publisher name appear repeatedly. This would have violated Second Normal Form and the solution would have been to remove the publisher information to a new table.

A.2.3 Third Normal Form: Eliminate Columns Not Dependent on Key

If a field does not help describe the key, then it needs to be removed to another table. Let's assume that we had a table of perhaps 10 books, and it just so happened that the publisher in each case was different. We might recognize the potential for violating the previous normalization rule (Second Normal Form) as we gathered more books, but beyond that whatever information we have about publisher, such as name and city, clearly is not part of the title, and thus should be removed to another table.

Although it can be argued that a particular title name may be the same from more than one publisher and that publisher really is required to describe the key[13], we'll assume that we have

[11] The intention is for most readers to examine and comment on the genealogical concepts presented in the data model rather than on the more arcane aspects of the data modeling methodology itself.
[12] This also violates Second Normal Form because the same author information for multiple titles is redundant.
[13] An example is "Fishes of the World", published by Grosset & Dunlap, New York, and the exact same title published by Halsted Press, New York. While publisher name distinguishes these two titles, so does author name. The former was written by Allan Cooper, and the latter by G. U. Lindberg.

a unique number like the ISBN as a unique key. Thus, we would want to remove the publisher information from this table.

Experience shows that inevitably if data is left in a table and violates Third Normal Form, some additional requirement in the future will cause that data to have to be broken out anyway.

A.2.4 Fourth Normal Form: Isolate Independent Multiple Relationships

No table may contain more than a single one to many or many to many relationship unless those relationships are themselves directly related.

Using our publishing example, a book might be available in more than one format, such as paperback, hardback, library edition, and collector's edition. So, we might create another table called FORMAT to store this information. A particular book might also be available in several languages, as are the Italian writer Umberto Eco's works. It might be tempting to stick the language field in the FORMAT table since one might think of a particular book as appearing in a particular language and a particular format, such as "The Name of the Rose" in an English version hardback.

However, the language and the format really aren't related to one another and thus putting them together in FORMAT violates Fourth Normal Form. The obvious problem this creates is that there may well be a German hardback and a French hardback for this title. The FORMAT table would eventually start filling up with all the combinations of language and format, and it would soon be apparent that the FORMAT table violates the Second and Third Normal Forms. Thus, independent multiple relationships should not be kept together.

If, on the other hand, we could imagine a world where all Italian books were issued only in the Quarto format, American books only came out in paperback, and all German books were in hardback form, then the two multiple relationships (books to formats and languages) would in fact *not* be independent, and they *could* be put together.

A.2.5 Fifth Normal Form: Isolate Semantically Related Multiple Relationships

There may be certain restraints on information that suggest separating logically related many to many relationships. This rule is most often effective when thinking about the number of rows required when adding and deleting data, but it tends to be highly context-specific and relatively rare. In our book example, we would have to postulate that the United Nations decrees that henceforth all books published in any language must be published in hardback and paperback and a library edition, and that any bookstore selling books must offer them in every language and every format. At that point, keeping format information in one table and language information in another table and relating them to each title becomes redundant. The original concept was that a particular book might only be available in a limited number of languages and formats, but now since every book is available in *every* format we can eliminate the many different combinations by simply storing languages in one table, and formats in another, and not relating them to an individual title at all.

Because Fifth Normal Form is not often a problem in data modeling, we have chosen to normalize only through the Fourth Normal Form, but we know of no Fifth Normal Form potential problems in the model.

A.3 THE ENTITY RELATIONSHIP DIAGRAM

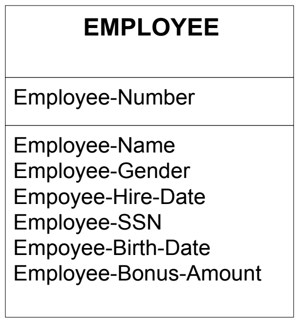
The terminology for the data model exists in what we call an Entity Relationship Diagram, sometimes called an ERD. ERD terminology expresses the logical equivalent of the same physical concepts as in RDBMS data storage. The purpose of using different names for the same concepts is to distinguish *abstract* or *logical* data modeling terms from *physical* RDBMS terms. To illustrate a proposed data modeling construct, we might wish to talk about what the data would actually look like in a table, and so we need different terms for the same ideas, depending on whether we're discussing the *logical* (abstract) or the *physical* (real tables). It's important to keep this distinction in mind, because our purpose is to understand and express the nature of genealogical data, not to actually construct a genealogical application. The physical representation of the logical model may be distorted for performance or other development reasons.

There are only three main pieces of the data model: entities, attributes, and relationships. An **entity** is similar to a table, an **attribute** is similar to a column or field, and a **relationship** expresses in both graphical and narrative form how groups of data are related. Note that we don't focus on rows or records in data models, except when turning a graphical relationship statement into an English statement, or when showing example data to illustrate the entity. We call this sample data row an **instance**.

A.3.1 The Entity and the Attributes

We can define an entity as any person, place, or thing (including events and concepts) about which information is kept. Entities, because of their definition, are named like nouns, with singular names like CUSTOMER or EMPLOYEE. An entity is a collection of similar objects called instances; in other words an entity, like a table, is made up of instances (rows or records).

Consider the following entity.



The box format shows the entity name at the top (EMPLOYEE), and it shows the attributes in the two boxes below that. The upper of these two boxes is reserved for attributes that are significant keys to the data. Keys are the attributes that either give the instances their unique identity, or the attributes that allow the entity to be linked in relationships with other entities.

A.3.2 Choosing Keys

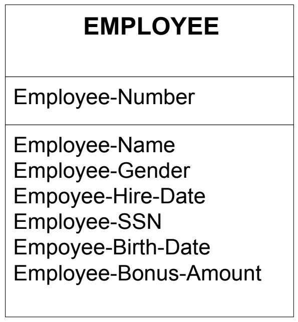
While it could be argued that we can simply identify attributes and let the developers decide which ones are the keys, in practice this doesn't work out very well, because it tends to mask problems that might exist in the model. The following guides may prove helpful in identifying keys; conversely, the inability to reasonably identify a key suggests that the model is not correct.

The most important characteristic of a key attribute is that it not change its value over the life of each instance of the entity. An instance takes its very identity from the key, so if the key changes, it's really a different instance. Because of the conflicting evidence typically found in genealogical data, most applications identify people by a unique (and unchanging) record ID that may be inherently meaningless, but which has the advantage of being stable.

The second characteristic of a key attribute is that it should be as small as possible. Although this rule is heavily influenced by physical database constraints (larger keys take up more space), it can be useful when the alternative is to either make up a small key, or use an awkward concatenation of multiple attributes to create a unique key.

The third characteristic of a key attribute is to avoid the use of intelligent keys, where the structure of the key indicates groupings. An example of this would be using a Henry number as a key to a PERSON entity; the obvious problem in genealogy is that an additional older sibling may be found subsequently, causing all the downstream Henry numbers to change. Note that this doesn't say that genealogists shouldn't use Henry or similar numbers; it just says that we shouldn't use them for key attributes[14].

Consider choosing a good primary key in the example EMPLOYEE entity.



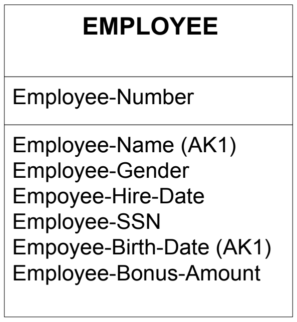
The diagram shows "Employee-Number" as the primary key, and this makes sense since employee number is unique for every employee. "Employee-Name" is not a very good candidate since if our company is very large we might have two John Smiths. "Employee-Name" could be combined with "Employee-Birth-Date" to make a better key, assuming that we don't have two John Smiths born on the same day. In general, these sorts of gee-that's-unlikely-to-happen solutions tend to lead to bizarre problems that don't turn up for years, specifically in this case until the second John Smith, born on December 12, 1962 is hired at the company. Who in the personnel department would know that some data modeler years earlier created a de facto rule for the company, that no two employees can have the same name and birth date? This hidden "gotcha" exists in many systems today; no matter how unlikely an event is, if you use the system long enough, probability theory says that the event *will* happen. This observation is undoubtedly echoed by experienced genealogists who have uncovered all sorts of unlikely events. Thus, we took extra care to avoid those sorts of problems in our genealogical data model.

[14] As a practical matter, Henry or other numbering schemes can easily be generated by the software application on the fly as needed in reports and book output. Consequently, a Henry or Register number is unlikely to be stored at all, unless the user interface suggests that genealogists prefer to search for data using a number like this; in that event, it would be necessary to store the number to facilitate indexed lookups, but the system would have to regenerate all the stored numbers if the tree structure is disturbed by adding, deleting, or modifying any of the family members.

"Employee-Gender" is also obviously not a good key because we will have many males and many females in the company. Similarly, there were probably many people hired on the same date, and even many employees that have the same birth date. "Employee-Bonus-Amount" is a terrible primary key because many employees will not be eligible for bonuses and so will have null (or empty) data; in addition, multiple employees may have the same bonus amount. None of these keys are very good as the primary key because they don't uniquely identify the instance (row).

"Employee-SSN" is an excellent candidate for a primary key since social security numbers are unique by definition. However, if our company hires any non-US citizens, they will not *have* a social security number. We were particularly careful to consider ethnic bias in building our genealogical data model, because an incorrect or biased model makes it difficult or impossible to enter certain kinds of ethnic genealogical data such as unexpected surname concatenations or even reverse order surnames.

Thus, in this example, "Employee-Number" was probably the correct choice for the primary key. Note that as data modelers we don't have to be concerned with additional indexes that allow the system to order the data in a particular way; that's a problem for the developers. In this example, there will undoubtedly be a report ordered by name, but whether a special index is built and maintained is a performance issue not related to data modeling. If it becomes important at a later time to remember that certain alternate keys need to be created, we can designate them with an alternate key number ("Akn") as shown below. Note that in this example, the alternate-key-number-one consists of both the name and the birth date, by definition in the order shown ("Employee-Name" + "Employee-Birth-Date").



Currently, there are no alternate keys in the data model.

A.3.3 Relationships

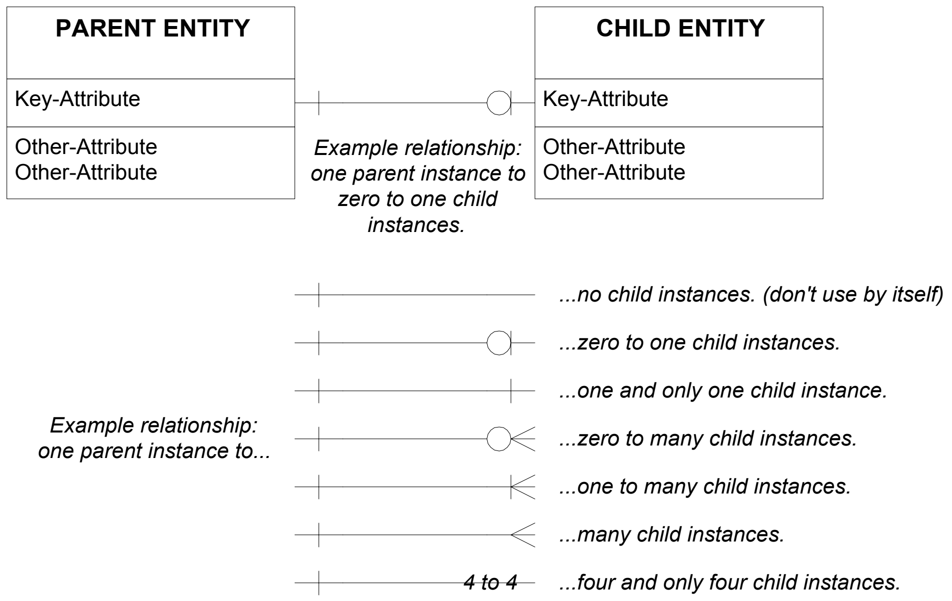
Relationships show the connections between entities, and can be read as verbs in relationship sentences that we can construct. Some examples include the following.

A PLAYER <plays on> one and only one TEAM.

A SALES-REPRESENTATIVE <sells> many PRODUCTs.

A DOUBLES-WRESTLING-MATCH <requires> exactly four PLAYERs.

Somewhat confusingly for our genealogical purposes, the entity on the "one" side of the relationship is called the parent and the entity on the "many" side of the relationship is called the child. Of course, when connections are one parent to one child, the concept of which entity is the parent becomes arbitrary and not very useful. The connections that we would typically use are shown below; note the specific example from the wrestling match (4 and only 4) that illustrates unusual conditions[15] that must be explicitly spelled out.


PARENT ENTITY
Key-Attribute
Other-Attribute
Other-Attribute
CHILD ENTITY
Key-Attribute
Other-Attribute
Other-Attribute
Example relationship: one parent instance to zero to one child instances.
Example relationship: one parent instance to...
...no child instances. (don't use by itself)
...zero to one child instances.
...one and only one child instance.
...zero to many child instances.
...one to many child instances.
...many child instances.
4 to 4
...four and only four child instances.

So, the single line means *one*, or if all by itself as on the left it means *one and only one*. The open circle means *zero*. The crow's foot means *many*. Special numbers are written on the flat line which by itself, as shown in the first example above as "no child instances", is meaningless.

The symbols used here were carefully chosen to be legible to beginning data modelers, although you should be aware that different data modeling methodologies use slightly different symbols. While "square crow's feet" from another modeling methodology are still quite understandable to someone familiar with the regular branched crows feet above, methodologies like IDEF that use closed and open circles become an exercise in frustration as the symbols have to be memorized because they have no intrinsic associative meaning.

[15] The rarity of these kinds of constructs suggests that we're unlikely to see this kind of condition in our genealogical data model.

Note that none of the conditions above are the same; zero to one is not the same as one to one because clearly the former case allows for *either* zero or one, while the latter means there will be one and only one. Similarly one to many does not mean the same as many; the latter means that there must always be *more* than one, i.e., two or more.

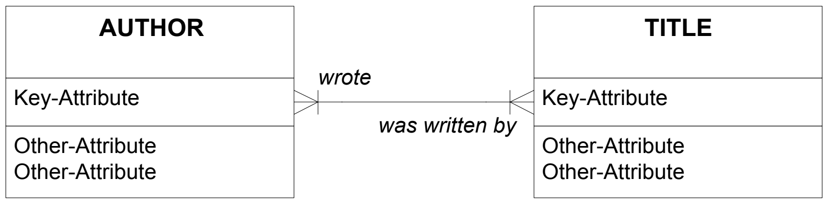
The differences between these multiple conditions may seem overly subtle or even trivial, but it can be extremely important whether it is possible to have one to many children or zero to many children. For example in genealogy, we might have one PERSON who has *one* to many NAMEs. In subsequent consideration, we might come to realize that we often know facts about a person, but do *not* know his or her name, so the correct model (in this example) would be one PERSON has *zero* to many NAMEs. It's an important distinction, because it means the software must allow the zero condition, and thus we know (if zero is the correct boundary condition) that name can *never* be used as a reliable person identifier because some instances won't have a name at all, while other instances will have multiple names. Of course, being genealogists, we already knew that person name was not very reliable as an identifier by itself.

In the relationship examples above, in every case the parent side of the relationship was one and only one; there was one and only one parent instance for differing numbers of children. But what about the relationship that we saw in the RDBMS discussion between books and authors on page 80? One author may have written a single book, while another author may have written multiple books, so clearly the TITLE side of the relationship is a one to many. We can say:

An AUTHOR <wrote> one to many TITLEs.

Unfortunately, while every book was written by an author[16], some books are written by multiple authors. Thus, we must also say:

A TITLE <was written by> one to many AUTHORs.



Note that the verb phrases have finally migrated onto the model, now that we have introduced the relationship line. There is a simple way to read the diagram. Starting with a line that runs left to right, the verb phrase *above* the line is read as [left noun <verb phrase> (number of) right noun], or "An AUTHOR / wrote / one to many / TITLEs." The verb phrase *below* the line is read from right to left, and when possible, is placed as close to the right side entity as possible to remind us that it follows the right side. Consequently, we say "A TITLE / was written by / one to many / AUTHORs."

The same rule applies to lines that are not horizontal. Imagine rotating the diagram above 90 degrees to the right (clockwise), so that gradually the AUTHOR entity moved *above* the

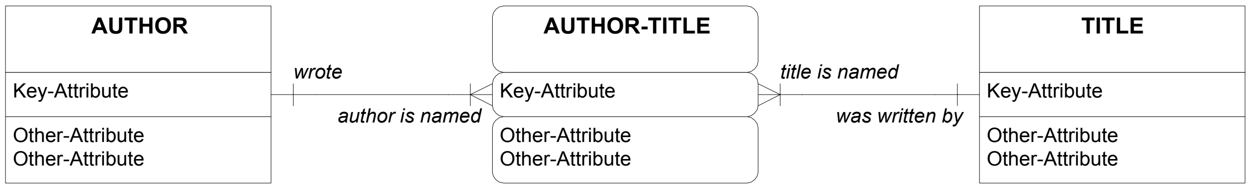
[16] In this example we'll ignore books published by institutions where, somewhat fictitiously, there appears to be no author. Obviously one or more real people actually wrote or compiled the book, but some publishing conventions suppress this information, leading to interesting problems in citing sources in an automated fashion.

TITLE entity. Disregarding the fact that we don't actually turn the entity boxes on their side, the relationship connector is now vertical, and the "wrote" verb phrase has moved in place to the right of the vertical line. So, we simply read down the right side of the line from the top entity to the bottom entity, and we get exactly the same verb phrase.

One last comment about verb phrases. Although the example above shows both verb phrases in the past tense, the general practice is to write all verbs in the present tense to be consistent. So, we might have "writes" and "is written by". As you may have noticed, usually one verb phrase is active ("wrote"), and by necessity the paired phrase is inactive ("was written by") because it expresses an entity acted upon rather than one that is acting on others.

More significantly, however, while the two statements are true, and the simple data model appears to be correct, *whenever you have a many to many situation, the model should be further broken down to remove all many to many conditions.* There are two reasons for this. First, if we attempt to put these many to many relationships into physical tables as we saw in our RDBMS table example, it results in coding problems. The second reason is that by decomposing the model to break up the many to many relationship, we *may* uncover additional useful information about the data.

Unfortunately, in order to break up a many to many relationship, we need to create a rather abstract entity that combines characteristics of both entities. These "made up" entities often cause beginning data modelers a lot of trouble because they're not rooted in the real world that the model is supposed to represent. That's one reason we displayed an actual bridging table (AUTHOR-TITLE) in the RDBMS example. The data model, though, might look like the following.



The new entity in the middle, AUTHOR-TITLE is unimaginatively named as a combination of the two entities that it artificially separates. Some data modelers go to a great deal of trouble to create new names such as BOOK or WORK, and while this has some merit, our convention here is to use artificial names for these artificial constructs.

Also note that the shape of the new entity is different; the corners are rounded to show that it's an associative entity. It depends on other entities for its identity, and has no meaning by itself. Although some methodologies are very strict about labeling dependent entities and drawing them with rounded corners, this is primarily so that rules about referential integrity can be stated. Referential integrity means that you don't delete the parent record if the child record depends on that parent record for its identity.

A.3.4 Entity and Attribute Definition

Every entity should be defined. An example, using a different data model, is: "A CUSTOMER is someone who buys something from our company." This is not a very good definition because we don't know whether a customer is really a *person* or whether it can be a corporation. We also don't know whether a CUSTOMER has to have already made a purchase from our company in order to be considered a customer. Are *potential* customers included? Thus, we want to make our definitions as precise as possible. This is particularly important with genealogical data since a lot of our data consists of evidence that is most certainly *not* “completely true facts” in the sense of our final conclusions. Most business applications don’t deal with nearly as much, if any, messy, conflicting data.

In the case of our associative entity, AUTHOR-TITLE, defining it requires more imagination. We might start by saying it’s “a book that someone wrote”. This is not correct, because AUTHOR-TITLE really does not consist of a list of books.

We might then define it as “information about books that people have written”, but again information about the books themselves, such as title, is recorded in the TITLE entity. The proper definition for this ugly entity is probably “Information about which TITLEs an AUTHOR has written and which AUTHORs have written a particular TITLE.”

The entity definition should include a description, examples, and any necessary comments.

Similarly, attributes should also be defined. This is particularly difficult with things like “Name” which can be defined many ways in general business applications, and even *more* ways in genealogical data. Without the descriptions, examples, and comments, however, any data model is difficult to understand.

Attribute definitions should also include domains, which means the values that a data element can take on. For example, if “Employee-Gender” is an attribute, then it would be logical that the allowable values are M (for male) and F (for female)[17]. These should be explicitly listed because they form an important part of the business rules that govern the data model. Note that this is only important where the data modeler feels that some standard *externally-viewable* codes are needed. The developers will likely have extensive *internal* codes that they use, but those are not of interest to us at the logical level.

Although human languages are filled with synonyms and homonyms, it is critical that every entity and attribute have one and only one name, and that we use these names consistently. When someone speaks of the attribute called “Name”[18], we must all know exactly what that person is talking about, and not guess. This, after all, was the purpose of the Lexicon project in the first place.

It should be noted that the description of the entity and the attributes is placed in a document called a Data Element Dictionary (DED) or simply a data dictionary.

[17] This presupposes that no data will be stored in any language other than English; M and F don’t work in all other languages, of course. This is an important consideration for a genealogical data model since genealogical data tends to be more international than in some traditional business settings. Even the traditional M and F choices are becoming somewhat riskier as transsexual biotechnology makes a continuum of what once was clearly binary, either/or, one or the other, data.

[18] Confusingly, the *name* of the attribute is the word “Name”, by which we mean a person’s name.

APPENDIX B: LOGICAL VIEWS OF THE DATA MODEL

Because the data model is somewhat abstract, it may be useful to relate the distribution of the data to certain concrete views, forms, or reports with which genealogists may be familiar. This section is not intended to be comprehensive, because our intention is to describe genealogical data the way it is used in genealogical research, not to design a computer application. But some sample views may prove helpful.

B.1 RESEARCH PLAN AND TASK LIST

The Research Plan contains data from RESEARCHER, PROJECT, and RESEARCH OBJECTIVE. It shows the research objectives for the particular project, perhaps arranged by researcher if there is more than one researcher. It may also show the scheduled SEARCHs. Additional features might include a statement of the SURETY system used for the PROJECT. Note that in a complex PROJECT, the RESEARCH OBJECTIVEs may be hierarchical.

A Task List is a special kind of Research Plan, and typically would consist of data from SEARCH for searches that have not yet been conducted, selected by some criteria such as REPOSITORY. Thus, the researcher might print a Task List of SEARCHs that she or he wants to conduct on a field trip to a particular REPOSITORY, knowing in advance what SOURCEs are available there and how those SOURCEs might be used for the researcher's needs. Note that an uncompleted SEARCH has a blank Completed Date attribute.

B.2 RESEARCH LOG

The Research Log is similar to the Research Plan discussed above in that it has data from RESEARCHER, PROJECT, and probably RESEARCH OBJECTIVE. But it also contains information about the SEARCHs actually conducted, and normally will contain more detail about each SEARCH such as the REPOSITORY and SOURCE. Note that the Research Log will display the attribute Searched For and the attribute Comments from SEARCH, both integral to understanding the research that has already been conducted.

B.3 CITATION

Citations are formed in a hierarchical fashion by taking the CITATION-PART from each level of SOURCE. The CITATION-PART is clearly labeled by CITATION-PART-TYPE so that the many CITATION-PARTs associated with a particular level of SOURCE are identified. If the researcher wishes to include the call number of the particular copy searched, then additional citation data comes from REPOSITORY-SOURCE and REPOSITORY.

APPENDIX C: DATA MODEL CONNECTIONS FOR EXPERT SYSTEMS

Several parts of the data model lend themselves to connection to an external expert system data repository, or authority tables as they're sometimes called. An expert system is one that contains a great deal of information about data in a particular category, without regard to the genealogist's particular data. For example, a great deal is known about person names such as given names and surnames. Some of this data can be gathered from large genealogical data repositories simply by counting unique occurrences. Thus a large genealogical database might yield 10 spellings of a particular surname. One difficulty with this approach, however, is that it may not be clear from the genealogical data submitted whether the surname spelling variant was actually the way the data appeared on a record or was an error made by the transcriber. Another approach, of course, is to gather names by carefully studying the evolution of particular names over time from a variety of sources. This is the approach used by people who specialize in this field, and who often publish surname or given name (often baby naming) books.

The group identified four areas where an expert system would prove useful: person names, place names, dates, and historic events.

C.1 PERSON NAME EXPERT SYSTEM

It is useful to know something about given names such as their origin and meaning, but for genealogical purposes this data may not be operationally useful. For example, knowing that "Robert" means "flame-bright" from the Old English *Hreodbeorht* and from the similar Norman and German names will likely not help one establish a genealogical connection. But knowing that Rob, Hob, Dob, Nob and Bob are all pet forms of the same name can be extremely useful, as is knowing that Gail is short for Abigail. And being able to connect the German Hans with the American John can be crucial for a genealogist to track a name change through immigration. The extent of information about given names, particularly across a variety of countries and ethnic groups, is massive and likely far exceeds any one genealogist's understanding of given names.

Data exists that shows the frequency of occurrence of given names, primarily from birth records in the 20th century. Although this is not helpful for earlier times, the Registrar General's Indexes of Births from England and Wales show that in 1900 Albert was used for 333 out of 10,000 male births while in 1990 it was down to 2 in 10,000.

Similarly, a great deal of information is known about surnames, including common spelling variations, and even the geographic and ethnic origin of these names. Thus, discovering a particular surname in a record can lead to an immediate understanding of the likely (but of course not guaranteed) ethnic origin of the name. Most American genealogists recognize a variety of surnames of English, Irish, Scottish, German, and perhaps French origin, depending on their own research interests, but most of us can not readily separate Dutch and German surnames, or identify Eastern European surnames.

Another useful attribute of surnames in an expert system is some indicator of the frequency of the names in given places and times. Thus, there are analyses that show the frequency of occurrence of various surnames in the 1790 census, for example. Frequency data can help a genealogist in determining the relative rarity of a particular name. And if massive genealogical repositories are combed for surnames, some indication of the most likely places to look for data can be gleaned.

In addition to surnames and given names, however, there are a great many other kinds of names, and a knowledge of name parts can be critical in genealogical research. The following table shows some name parts, but it is not intended to be comprehensive. A name expert system would be able to recognize name parts by their position or by looking them up in tables extracted from large databases of names, or from lookup tables prepared by subject matter experts.

Note that some of these name parts, such as patronymics, may be identical with other name parts such as surnames. Thus, Anderson is both a patronymic and a surname, and no doubt somewhere it has been used as a given name as well, and perhaps even a religious name.

PERSON NAME PARTS

Number per person	*Name Part Description*	*Example(s)*
Many	Title/prefix/prenomial[19]	Dr.
Many	Personal name/mononame	Joe, Joseph, Susan, Sitting Bull
Many	Family/clan name	Jones, McGregor
Many	Postnomial	Jr., VIII
1	Religious name	Mary Patrick
Many	Nickname	Skip, Rusty
1	Dit name	Fontaine/Bienvenue/Welcome
Many	Connector/article	De la, van
1	Infix	John **Cardinal** Cushing
1	Locality name	Ash, Attlee, Byfield, Uphill
1	Occupation name	Smith, Bishop, Taylor
1	Patronymic	Ericson, Ivanovitch
1	Metronymic	Leurunessone

C.2 PLACE NAME EXPERT SYSTEM

Place names are not only hierarchical in nature, with cities inside counties, which are then inside states, which are then inside countries, but there are many kinds of hierarchies depending on types of jurisdiction and location. Worse, places are known by different names from *other* places (the German word for the country the English call France is not the same name as the French call their country), and names and boundaries have changed over the years.

An expert place name system recognizes alternate names for the same place, the dates associated with the formation of that place and in the case of obsolete place names, the date associated with the end of that place name. Further, the expert system recognizes the boundaries of places, perhaps as vector data or as points on a grid, and the expert system can calculate the distances between point-data such as the distance from the center of one small town to another.

The following is a list of issues related to place names.

- A place name must be captured in its entirety exactly as written.

- Place names can nest, with small places located within larger places. There is often a hierarchy for Place Names such as town within county, county within state, state within country.

[19] It is our understanding that the term "prefix" is used by linguists to mean the first part of a name part, not the first part of a whole name. The term "prenomial" is used for that instead.

- There are different names for the same place levels. At approximately the same hierarchical level, one system may include a *province* name while another includes a *state* name.

- Place names include overlapping areas. Not all areas are strictly nested, particularly when comparing areas either over time, or across different jurisdictional frames.

- Time is important to place names and must be linked. A place name does not mean anything without a time reference; for example, where are the boundaries of Austria, and is a particular town inside Austria or part of some other country? The answer depends on the date associated with the place.

- Place names may refer to a geographic feature. Place names refer to farms, towns, and larger legal constructs, but they also apply to geographical features like mountains and lakes.

- Place names may be political. We think of political divisions with respect to many names such as a particular county.

- Place names can be jurisdictional. Many place names are not political, but are part of other jurisdictions such as religious frameworks.

- Place names can be referential. Some place names are not explicit, but are merely referential. "Place of his birth" might be a significant piece of place data, but it is not an explicit place name; it potentially points to a place name, however.

- Places have boundary data. Place names generally refer to two dimensional objects, but these objects may be irregular polygons (e.g., county), or when seen from a sufficiently high level they may be linear (e.g., river or trail). Boundary data needs to exist against a known grid system to be meaningful. See the item below.

- Places have spatial coordinates. Even place names that are, in one particular view, simply point data rather than two dimensional data, must exist on a spatial grid and thus have associated with them spatial coordinates.

- Place names evolve over time. Clearly the *boundaries* of place data change over time, but even the *name* of a specific place may change over time, e.g. St. Petersburg and Leningrad.

- It may be useful to think of places as having genealogies of their own, with multiple parents, a birth or creation date, and a death or dissolution date.

- Place data, like date data, is not intended to be limited in range and representation.

C.3 DATE EXPERT SYSTEM

In order for a date to be meaningful, it must be understood in the context of the place where it was used. This means the genealogist must understand the calendar system in use in that place at that time. A date expert system, although it could be set up as a series of tables like the name and place expert systems, will understand dates associated with places and be able to freely calculate between different calendar systems. Thus, the expert system has tables that indicate such things as the start of the year (January 1st, March 25th, etc.), the calendar algorithm system (Julian, Gregorian, Islamic, Hebrew, Quaker, French Republican, etc.), and any anomalies about the calendar in that place (the year 1700 in Sweden, etc.). Dating systems that involve feast days and other difficult to calculate events should be available so that the genealogist can determine exactly when, in the Julian calendar for example, a particular feast day in 1737 occurred, or when that day occurred in, say, our modern Gregorian calendar if he or she wishes to compare it to another date.

The following is a list of issues related to dates.

- Time is continuous. This means that an appropriate level of granularity, such as minutes versus days versus decades versus centuries must be chosen.

- Time is innumerable. There have been many days in history and the list continues to grow, daily as it were.

- Dates exist in multiple calendar systems and require different algorithms to compute. There are many calendar systems that have been used, such as the Julian and the Gregorian, and there have been many variations of most calendar systems, such as the Swedish attempt to convert between the two.

- Some systems require table lookups because of their irregularity; they are too unusual to allow computation through an algorithm.

- Dates may only be partial dates. A date may include the day, month, and year in a particular calendar system in a particular country, but a date may only include the month and year, or even the year alone.

- Dates are associated with events.

- Dates are associated with places.

- Date data, like place data, is not intended to be limited in range and representation.

- Dates are associated with particular cultures, such as Quaker dates.

- Dates are ambiguous, soft data. Examples include dates marked abt, ca, est, say, ≤ (less than or before), or ≥ (greater than or after). Soft dates are also expressible with a ± range.

- Ranges may be appropriate for date data. Not all events or places are point data with respect to dates. For example, a single date is not appropriate to mark a trip that might have lasted months. Other dates cannot be fixed any more precisely than a range. A date that happens between two hard dates is itself soft.

- A time span may be required for date data. Two dates can be used to mark the span, or a single date plus duration, e.g., “four years military service from 01 Jan 1862”.

- Date data can be relative. Dates can be relative to an event, such as “four months after marriage”.

- The sorting of dates is a non-trivial matter, since decisions must be made regarding how to sort records with soft dates or ranges, with records that have hard dates.

The group identified the following as the purpose of dates in genealogical research and analysis.

- To sequence events;
- To calculate the occurrence of an event;
- To match items; and
- To calculate the duration of an event.

Dates do not appear in any entity designed to hold solely dates for several reasons. First, while dates could be stored in a table format, that is, a table of days in a particular calendar system, the number of records would be quite large and would continue to grow by one record per day. Secondly, the nature of dates and calendar systems is that, with some exceptions, dates in most calendar systems can be determined algorithmically by calculating in a standard (if complex) fashion from a known starting point.

Finally, as indicated in this section, more complex date handling is the subject of advanced or expert systems outside the scope of this initial model. Thus, while date is an attribute throughout the data model, it does not appear in any sort of lookup table or authority of dates.

C.4 EVENT EXPERT SYSTEM

The purpose of an event expert system is to allow the researcher to place research data in the context of known significant historical events.

An event expert system contains data about known historical events such as wars, science, technology and inventions, music, political events, birth and death days of famous figures, and so forth. Some event data, such as the formation of political states, is already considered in the place expert system, but almost all other event data is related to place as well.

In fact, an event expert system needs to understand the hierarchy of place in order to be able to select relevant events against the genealogist's research data. For example, if the researcher has a birth date and a farm location in the Oklahoma Territory, an event expert system should be able to supply historically significant facts at the local level (to some point), the territory level, the United States at that time, and the rest of the world.

C.5 OTHER EXPERT SYSTEMS

Virtually all the attributes of ASSERTION can have authority tables or external expert systems. In addition to person name, place, date, and event authorities, tables can be created for occupation names, descriptions, and locations where found if appropriate; relationship names including degree of consanguinity in various systems; and so forth.

APPENDIX D: ADDITIONAL STATEMENT TYPES

Section 3 discusses the original three genealogical statement types and the final Super Statement type that they evolved into. This appendix discusses two other statement types that the group considered and then ultimately rolled into the Super Statement type along with the three original statement forms. We include this material for readers who may be interested in the other intermediate forms.

D.1 Statement Type 4: Statements About Sequence

The fourth statement type expresses the sequencing of data, and this statement has the following general forms.

Event 1 / Order / Event 2.

An example of this would be an authoritative county history that tells us that the Sherman Oaks Road was built after the Flood of '77 washed away the old road, which was called the Denison Post Road. The name of this road helps us place undated documents from this area. Two statements of the sequencing form appear as follows.

Sherman Oaks Road built / after / the flood of '77.

Sherman Oaks Road built / after / Denison Post Road destruction

Another subtype of this sequencing statement form is the following.

Characteristic 1 / Order / Characteristic 2.

This subtype allows us to sequence characteristics like occupation. A sample statement is the following.

Was a butcher / after / was a teamster

A final subtype allows us to sequence the order that a person was called by two names. The form is the following.

Name 1 / Order / Name 2.

An example of this type of sequence statement is the following.

Was called Butch / before / was called Skip

D.2 Statement Type 5: Statements About Rank In A Group

This last type of ordering statement is required for certain kinds of evidence such as birth order: "was the second son". Although this particular statement of birth order has multiple interpretations,[20] it is important that we be able to capture statements of the following form.

X / Rank / Group

John / 2nd son / union of Robert and Mary

In this case, "X" stands for any type of genealogical data, e.g., person, characteristic, relationship, event, or place. Once this is evident, is can then be extended to groups themselves, so that a group is part of a larger, possibly ordered, group.

Many more examples come to mind. Evidence related to an ancestor that states "when John was a private" can be inferred to have occurred before evidence that states "when John was a corporal", although of course it's possible that John's military career took some unfortunate turns and he was busted back to private after he was a corporal.[21] Thus it might be useful to have a table of military grades and ranks appropriate to the service, so that, for example, one knew what rank a captain was, relatively speaking. And of course these grades and ranks are not the same in different countries, in different centuries, or even between different branches of the military in the same country.

If you examine the statement type "X / Rank / Group", you'll see that it can be used to express *all* the other sequencing statement forms, such as "Name 1 / order / Name 2" if there is an appropriate group. Thus a statement like "was called Butch / before / was called Skip" can be translated into the following two statements:

Name 1 / rank number 3 / group of names that this person had
Name 2 / rank number 7 / group of names that this person had

Although this seems somewhat awkward, it is a much stronger form and allows us to group and order any kind of genealogical data. In the example above, the initial rank might have been 1 and 2, but subsequent data allowed us to insert other names and thus recognize a whole series of names and nicknames that were used.

Note that equal rank implies equal identity in the context of the ranking system. Thus a modern Army Colonel and a Navy Captain are of equal (identical) rank[22], albeit in two different ranking systems subordinate to an overall military scheme.

The concept of a group within a group is useful for a group of 7 families in Montgomery County, New York that can be compared to a similar group of 7 families in Portage County,

[20] Examples: "was the second son born to this man and woman although the man had other sons"; "was the second son born to this man and woman although the first son died in infancy"; "was the second son who actually lived, two others having died in infancy", etc. Similar problems exist when we have statements like "older brother", implied order from census records that possibly show head of household, spouse, and children in order by age, or statements about the rank of children or wives in some Asian data. But sequencing can also apply to geographic data where "north of" specifies the relation of two places, or chronological data where one event happens before or after another.

[21] A more interesting and perhaps more common exception would be where John was a private in the New York militia after he was a corporal in the Connecticut militia.
[22] Rank in the sense of GROUP, but perhaps more properly "grade" in the military sense.

Ohio. In this case the researcher may be asserting that they're equal, but if there was a much larger community of families in Ohio, the New York group might be one of several groups combined into the larger group.

D.3 The Partially Combined Statement Type

There is one thing missing from the "X / Rank / Group" statement form, however, and that is date and place. In some cases we will only be able to say that the statement is true for a particular date and place. Thus, in a letter dated August 12th, 1856 from Philadelphia we learn that her intended "was called Butch before he was called Skip." Thus, the statement type becomes:

X / Rank / Group / Date / Place

Since as we indicated earlier, "X" can stand for any type of genealogical data, clearly we have come very close to replacing statement types S1 through S3, although we cannot make a statement of the form S1 because we do not have two person fields for the relationship.

Resolving these issues, however, we have the completely combined statement form, what we call the Super Statement Form. This is discussed in Section 3.