Florida International University
FIU Digital Commons

FIU Electronic Theses and Dissertations | University Graduate School

7-21-1998

A sem-odb application for the western cultures database

Raquel Ghersgorin
Florida International University

DOI: 10.25148/etd.FI15101565
Follow this and additional works at: https://digitalcommons.fiu.edu/etd

Part of the Databases and Information Systems Commons, and the Software Engineering Commons

Recommended Citation
Ghersgorin, Raquel, "A sem-odb application for the western cultures database" (1998). *FIU Electronic Theses and Dissertations*. 3926.
https://digitalcommons.fiu.edu/etd/3926

This work is brought to you for free and open access by the University Graduate School at FIU Digital Commons. It has been accepted for inclusion in FIU Electronic Theses and Dissertations by an authorized administrator of FIU Digital Commons. For more information, please contact dcc@fiu.edu.

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

A SEM-ODB APPLICATION FOR THE WESTERN CULTURES DATABASE

A thesis submitted in partial satisfaction of the

requirements for the degree of

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

by

Raquel Ghersgorin

1998

To: Dean Arthur W. Herriott
College of Arts and Sciences

This thesis, written by Raquel Ghersgorin, and entitled A SEM-ODB APPLICATION FOR THE WESTERN CULTURES DATABASE, having been approved in respect to style and intellectual content, is referred to you for judgement.

We have read this thesis and recommend that it be approved.

David Barton

Nagarajan Prabhakaran

Naphtali Rishe, Major Professor

Date of Defense: July 21, 1998

The thesis of Raquel Ghersgorin is approved.

Dean Arthur W. Herriott
College of Arts and Sciences

Dr. Richard L. Campbell
Dean of Graduate Studies

Florida International University, 1998

I dedicate this thesis to my husband Saul, whose love and support give me the strength to accomplish all my goals, and to my family who make possible my wishes and my dreams.

AKNOWLEDGMENTS

I wish to thank every member at HPDRC, who made possible this project, and all the FIU professors and friends who helped me through my learning process at this fine institution.

I also wish to thank the members of my committee, for their.

ABSTRACT OF THE THESIS

A SEM-ODB APPLICATION FOR THE WESTERN CULTURES DATABASE

by

Raquel Ghersgorin

Florida International University, 1998

Miami, Florida

Professor Naphtali Rishe, Major Professor

This thesis presents the evolution of the Western Cultures Database. The project starts with a database design using a Semantic modeling, and continues with the implementation following two techniques: a Relational and a Semantic approach. The project continues with them in parallel, reaching a point where the Relational is left aside because of the advantages of the Semantic (Sem-ODB) approach.

The Semantic implementation produces as a result the Western Culture Semantic Database Application – web interface (the main contribution of this thesis). The database is created and populated using Sem-ODB, and the web interface is built using WebRG (report generator), HTML, JavaScript and JavaChart (applets for graphical representation). The resulting semantic application permits the storage and retrieval of data, the display of reports and the graphical representation of the data through a Web interface. All of these to support research assertions about the impact of historical figures in Western Cultures.

TABLE OF CONTENTS

CHAPTER

I. INTRODUCTION **1**

II. THEORETICAL BACKGROUND **5**

The Relational Data Model 8
The Normalization Process 10
Advantages of a Relational Data Model 12
The Semantic Data Model 12
Advantages of a Semantic Data Model 14
Model Conversion Methodology 16

III. EVOLUTION OF THE WESTERN CULTURES DATABASE PROJECT **18**

High Performance Database Research Center 18
FIU President's Office 18
Western Cultures Database application 19
Initial Approach: an Analysis phase 19
Design phase 20
Data Collection 29
Implementation phase 30
Relational Implementation approach 31
Semantic Implementation (Sem-ODB) approach 37

IV. THE WESTERN CULTURES SEMANTIC DATABASE APPLICATION - WEB INTERFACE **44**

Tools used for building the application 44
How to start the application 46
The "Western Cultures Semantic Database Application" - web interface 46
Advantages of a Semantic Implementation approach 121

V. CONCLUSION **122**

VI. FUTURE WORK **123**

LIST OF REFERENCES **124**

APPENDICES **126**

I. INTRODUCTION

A few decades ago, the database technology emerged as an essential tool for data management, uniform access and control of consistent information retrieval. A database management system (DBMS) uses a data model as its underlying structure, to embody the relationship between its entities. The data base model specifies real world concepts in a form understandable by the DBMS, categorized as [Card&McLeod-90]:

- **Record-Oriented Database Model**: Common use of the record construct as foundation of the model, including the Hierarchical Data Model, the Network Data Model, and the Relational Data Model. The Relational in general, is the most common, viable and practical tool, commercially speaking.
- **Object-Oriented Database Model**: Common use of the object construct as foundation of the model, including the Object-Oriented Data Model, and the Semantic Data Model. This generation of database systems is growing, due to new demands in capabilities beyond those presented in the relational technology. The Semantic approach attempts to provide a rich set of modeling constructs, which are expressive of data meaning, and enclose the way users think about data. It offers a simple, natural, implementation-independent, and flexible specification of information. For these reasons, the Semantic methodology is expected to become predominant in future databases.

The High Performance Database Research Center (HPDRC), a research division of the Florida International University School of Computer Science, conducts research on

database management systems and various applications, leading to the development of new types of database systems and refinement of existing database systems.

Recently, HPDRC committed to developing a Western Cultures Database application for the FIU President's office. This application is required to store data about persons considered to be historical figures (according to book references collected by the office), due to their contributions in a specific field of work. Moreover, the application should generate reports and graphics, to support research assertions about the impact of historical facts in the evolution of western cultures.

This thesis presents in detail, the evolution of the Western Cultures Database project, and produces as its major contribution, a SEM-ODB application for the Western Cultures Database, presented on a Web interface.

The thesis includes a theoretical background that explains the different Data Base Models (Chapter II), focusing more on details about the relational and semantic methodologies.

The main difference among database models, lies in the representation of the relationships between entities, and this thesis presents a comparison pointing out the advantages of the Semantic model over the Relational one, to theoretically support the Semantic preference for developing the application.

After the necessary theoretical basis, this documentation includes an extensive and detailed explanation on the evolution of the Western Culture Database project (Chapter III).

Initially, because of the impact of Semantic modeling, the database was designed and modeled using this methodology, and represented by a semantic schema.

The first approach for the implementation of the project was to consider a Relational database management system, making necessary then a model conversion to obtain the design of relational tables. The tool used for implementing data entry is Oracle. Also Oracle graphics and Oracle reports are considered to cover requirements of data retrieval and presentation. However, at present, implementing the application using this approach, does not permit the user to take advantage of the Internet.

The second approach was to use a Semantic database management system, because all the capabilities of a Relational approach can be implemented, and also new features can be included (Web based front end) requiring less effort and time for development and maintenance. The Semantic/Object Oriented tool available at HPDRC (Sem-ODB) is a semantic object-oriented database management system interface, in C++ that permits the construction and usage of a database, based on the Semantic model.

This thesis starts the project using the two implementation approaches, and continues with them in parallel, reaching a point where the Relational one is left aside, and the Semantic one goes beyond producing as a result the Western Cultures Semantic Database application.

The Western Culture Semantic Database Application – web interface is the main contribution of this thesis and it's explained in detail in the following chapter (Chapter IV). The application permits the efficient storage and retrieval of data, and the useful graphical display of the data through a Web interface.

Finally, this thesis presents a conclusion (Chapter V), and suggests future work (Chapter VI) to improve the study as well as the Western Cultures Semantic Database application. The resulting code for each part of the project is included in the Appendices.

II. THEORETICAL BACKGROUND

" We must understand our past to manage our present and face our future" [Stevens-94]

The history of database modeling started with **Record-Oriented Database Models**, more specific with the **Hierarchical Data Model**, when programmers observed that many databases consisted of files hierarchically related. A file higher in the hierarchy is a parent file to the files below it. Subordinate files contain child records, and a child record may have children of its own. In this model, a child record type may have only one parent record type, and each child record instance may have only one parent record instance.

A hierarchical data model always starts with a root record, and every record consists of one or more attributes describing the entity at that record. Furthermore every record has to be accessed throughout its parent record, making the access path to every one (within a hierarchical data model) a unique path. **Figure 2.1** shows the components for a hierarchical data modeling.

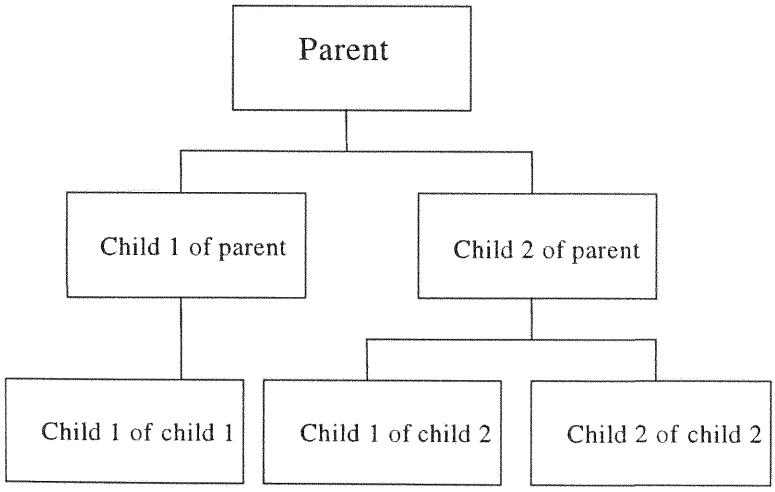


Figure 2.1 Components for a hierarchical data modeling

The relations supported by the hierarchical approach are the *one-to-one* relation, and the *one-to-many* relation. **Figure 2.2** shows how to model a *one-to-many* relation.

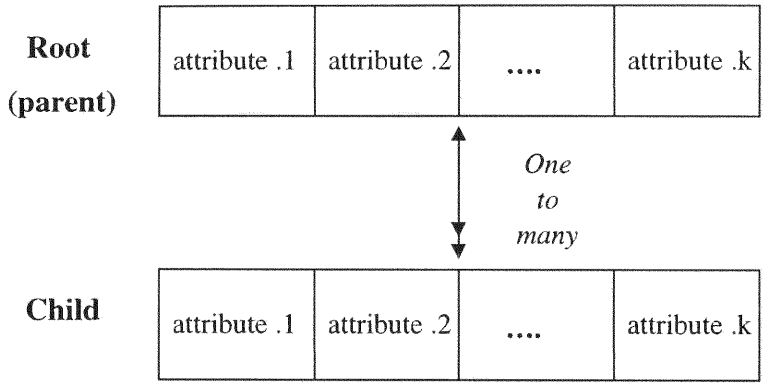


Figure 2.2 Modeling a *one-to-many* relation on a hierarchical data modeling

In this model, there is no efficient way to represent a *many-to-many* relation, being the only way to introduce two hierarchical data models with a relationship established between them. But then redundant data is stored, creating possible storage and update anomalies. Moreover, using that idea makes it difficult to insert and delete data.

Because the hierarchical data model does not support child nodes having more that one parent (a *many-to-many* relation), the **Network Data Model** evolved to support such requirements.

The major components of a network data model are sets of records inter-related by arrows, forming a directed graph of records. **Figure 2.3** shows the components of network data modeling.

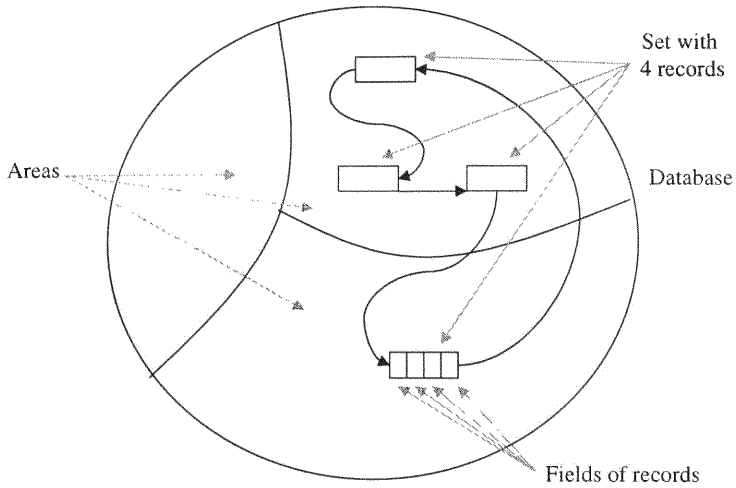


Figure 2.3 Components of network data modeling [Atre-80]

An area is a named subdivision of a database and may contain occurrences of records and sets or parts of sets. Areas can be operated independently or in conjunction with other areas. A set is a named group of related records, where there is only a single "owner" occurrence in a set occurrence, and there may be zero, one, or many "member" occurrences in the set.

Because of the set concepts, the *many-to-many* (and of course *many-to-one*) relations can be implemented easily using the model, but this complicates the application design and its programming. The programmer must be very familiar with the logical structure of the application which manages the database.

The previous two models support file relationships by embedding record pointers in the data records, generating in some cases hidden pointers in the data records [1], which

[1] The sets can overlap causing inheritance errors.

compromises the data integrity and, therefore, makes the database unstable. To solve this situation, the Relational Data Model emerged, representing the relationships between files by data value rather that record addresses.

The Relational Data Model

A relational database is simply a database that is perceived by the user as a collection of two-dimensional tables, where each table represents an entity or a relation of fields (stored in the columns) and records (stored in the rows, or tuples) [Date-86].

The notion of using a relational data model as underlying structure of a data base management system was introduced by Dr. E. F. Codd [Codd-70] in 1970, characterized by the following features [Codd-82]:

- The data structures are simple: Using tables facilitates the representation of data entities, because they can be viewed as files, whose elements are data instances. A relationship linking two tables is specified by a common attribute (field) in both tables.
- The data definition, retrieval, and update are facilitated by a set of operators (relational algebra operators): select, project, product, union, intersection, difference, join and divide. See APPENDIX A (theory appendix) for a Relational Algebra Operators explanatory figure.
- A set of integrity constraints defines consistent states of the database.

A system is said to be relational if it satisfies the following conditions [Vald&Garad-89]:

- All information in the database is represented as values in tables.

- No inter-table pointers are visible to the user.
- The system must support all the relational algebra operators, without any restrictions due to implementation.
- Two basic integrity constraints must be supported:
    1. Each relation has one or more attributes that constitute a unique key. An important consequence of this is that no attribute participating in the primary key is allowed to accept null values (Integrity rule called Entity integrity). This key also must be time-invariant.
    2. An attribute value that represents a relationship between two tables, must appear in both related tables. In those tables, that common attribute must be of the same value, or must be wholly null. (Integrity rule called Referential Integrity).

Based on the theoretical background and in programmer's experience, experts in designing Relational databases have encountered several problems [Ullm&Wido-97]:

- Redundancy: Information can be repeated unnecessarily.
- Update Anomalies[2]: On a database update, the information in one tuple may be changed, but in other tuple may be left unchanged.
- Deletion Anomalies: If a set of values becomes empty, other information can be lost as a side effect.

[2] Anomalies: Problems occurring when a database designer tries to cram too much into a single relation.

The way to solve these anomalies is to decompose relations using what is called normalization; its goal is to replace a problematic relation by several that would not exhibit the anomalies.

The Normalization Process

The Normalization process is a step-by-step reorganization of the elements of the data into affinity groups, by eliminating dysfunctional relationships and by assuring normal relationships [Koch&Loney-97].

The first step in normalization consists of transforming the data items into a two-dimensional table, which includes the removal of repeated occurrences of data items. The second step is to state what the keys are, and to relate the data items to the keys. The third step is to separate out data items from the second normal relations that, while dependent only on the keys, may have an independent existence in the database.

More formally [Atre-80] and [Ullm&Wido-97]:

- First Normal Form (1NF): A relation in the first normal form is a "table". At every intersection of a tuple (row) and a column there can be only one value in the table.
- Second Normal Form (2NF): A relation is said to be in second normal form, when every nonkey attribute is fully functionally dependent on the primary key (every nonkey attribute needs the full primary key for unique identification). Every relation in the second normal form is also in the first normal form.

- Third Normal Form (3NF): A relation is in third normal form, if there is no transitive functional dependency between the nonkey attributes. When a nonkey attribute can be determined with another nonkey attribute, there is said to be transitive functional dependency between the two. Then, a relation is in 3NF if every value is a fact about "the key, the whole key, and nothing but the key". Every relation in the third normal form is also in the first normal form.
- Boyce-Codd normal Form (BCNF): A relation is in BCNF, if there is no transitive functional dependency upon any key attribute. BCNF implies 3NF.
- Fourth Normal Form (4NF): The fourth normal form is essentially the BCNF condition, but applied to avoid multivalued dependencies instead of functional dependencies. A multivalued dependency is a statement about a relation, that when the values of a set of attributes are fixed, then the values in certain other attributes are independent of the values of all the other attributes in the relation.

As mentioned above, the purpose of normalization is to avoid possible anomalies in a Relational Database.

Advantages of a Relational Data Model

In a relational data model, the user is presented with a data model, where the requests are formulated in term of content, and not in term of physical implementation (as the previous models). This model does not reflect any storage structure or any low-level access strategy, providing relatively a higher degree of data independence. Because there is no positional dependency between the relations, requests do not have to reflect any preferred structure. Finally, because of the mathematical theory supporting the relations, this model has a solid foundation, assuring a rigorous method of designing a database. These days, data base management systems based on the relational model, are commercially available, showing the methodologies impact. However, in a following section, the Relational model will be compared to the Semantic model and those advantages will be carefully reviewed, supporting the use of the second database methodology for the Western Culture Database project.

The Semantic Data Model

The Semantic Data Model is an attempt to provide a rich set of modeling constructs, which are expressive of data meaning, and enclose the way of the user thinks about data. It offers a simple, natural, implementation-independent, flexible, and non-redundant specification of information; also, this methodology is expected to become predominant in future databases, replacing the current relational database methodology.

More formally, in the semantic model, the information is represented by logical associations (relations) between pairs of objects and by the classification of objects into

categories. These categories do not need to be disjoint, which means that one object may belong to several of them. The categories are preserved in time, and the objects may come and go. The object then represents a real world item, and it can be concrete (a printable item, with a value associated, for example the **name** "Venezuela") or abstract (a tangible item or an idea, for example the **country** associated with the name "Venezuela").

To represent graphically a semantic database (its structural description), designers use a schema that follows formalisms represented in **Figure 2.4**.

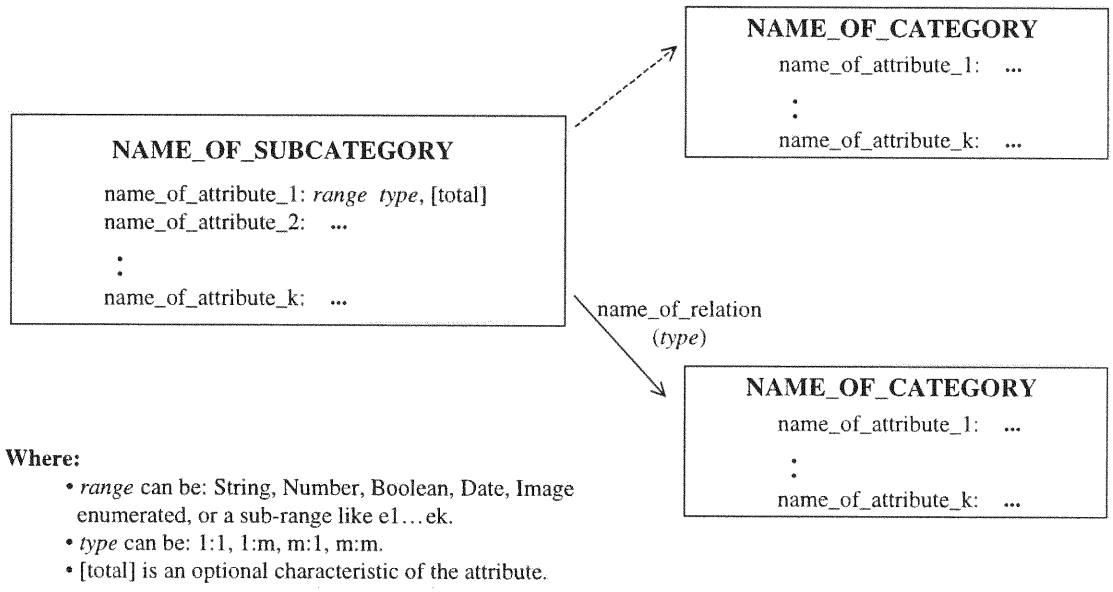


Figure 2.4 Graphical formalisms in a Semantic schema.

In **Figure 2.4** are summarized the schema formalisms, where categories are shown by rectangles. Relations from abstract categories to concrete categories (attributes) are shown inside the boxes, following the format: relation name: *range type*. Arrows show

relations between abstract categories, and the name and type of the relation are indicated. The types of relations are: m:m, m:1, 1:1, where **1(m) : 1(m)** represents that one (many) object(s) can be related to one (many) object(s). By default, the type of an attribute is m:1. Also a [total] suffix on an attribute means that every object on the category should always have a value for that attribute. Finally, the figure represents sub-categories (inheritance relations) by dashed arrows [Sem-ODB-98] and [Rishe-92].

Semantics models encapsulate structural aspects of objects, being it the reason why they are categorized as **Object-Oriented Database Models**; an Object-Oriented schema must capture not only the structural properties of a real world application, but also it must encapsulate behavioral aspects of the objects. However, the behavioral properties of the OO databases can be easily added to the semantic model when necessary.

Advantages of a Semantic Data Model

Logically organizing data with a Semantic database model facilitates the design, access and evolution of a database.

- Facilitating the design: A designer can organize a Semantic database by directly specifying his knowledge of the application environment, and because it's simplicity, the owner of the database (client) can also participate and understand the design. In the other hand, the mapping of an application environment into low-level and record-oriented structures, implies modeling limitations, and inevitably causes a loss of information (and misinterpretation later in the data) by expressing only a limited portion of the designer's knowledge of the application environment.

Additionally, semantic integrity constraints must be defined and enforced externally of the database structure [Yao-85]. Furthermore, mapping an application environment into record structures is a tedious task requiring a significant amount of software-engineering knowledge, and experience (the normalization process is a difficult methodology and an unnatural way of representing data).

- Facilitating the access: In a Semantic data model, is presented in a simple data model that permits the reference of a concrete object and the navigation through the schema by applying attributes directly to that object. In the Relational model, on the other hand, the user must navigate through the schema within the provided record structure, simulating pointers (keys) by comparing identifiers in order to transverse from one relation to another[3] [Card&McLeod-90]. In a Relational database, a key is a required field of a record, and it's value should not change in time (arbitrary changes may cause inconsistency of the Relational database); but, in the real world, keys exist very rarely because all natural attributes should be allowed to change in time: if not because of the changes in the real world, then, because of correction of erroneously recorded data [Rishe-93].
- Facilitating the evolution and permitting relativism: A Semantic schema is easier to modify than the conventional one, when the nature of the application environment activities changes. Also, a Semantic schema allows the structure of a database to support relativism: alternative ways of looking at the same information. On the contrary, in conventional models, it is generally necessary to impose a single

[3] Typically using the Join oprerator

structural organization of the data, carrying along a particular interpretation of the data, not suitable for all the users of the database [Yao-85].

Additionally, a Semantic model permits viewing and accessing the database at different levels of abstraction, going from the general view to a different levels of detail.

The comparison of the Relational and Semantic methodologies is clearly expressed by the following citation: "The relational databases have provided a good service in many conventional database applications. However, in situations where the structure of information is complex, or where greater flexibility is required (objects with unknown identifiers, or objects moving from one category to another, etc.), or where non-conventional data is involved (long texts, images, etc.), other approaches need to be considered: semantic and/or object-oriented databases." [Rishe-98]

Model Conversion Methodology

A binary schema is called table-oriented or a relational schema if:

1. All the abstract categories of the schema have keys.
2. All the abstract categories are pairwise disjoint.
3. The only relations between categories are represented by attributes.

Obtaining a Relational Schema from a Semantic Schema can be achieved following a top-down methodology that ensures the rules presented above:

1. Add a key for all categories: A time-invariant attribute of a category is called its key if it is 1:1 and total (single-attribute key)[4]. Also, exist cases where a minimal collection of attributes is sufficient to identify every object of the category[5]. Furthermore, to serve as a key, a new artificial attribute can be generated.
2. Eliminate sub-categories (preserve disjoint sets): It can be achieved by Union the subcategories in one category and inserting to it Boolean attributes that represent how the object will be categorized. Also, to eliminate sub-categories, the parent category can be eliminated, adding all it's attributes (including the key) to each one of the children, converting them into artificially disjoint categories of Events. Moreover, a conversion of Union+Events can be applied, where all the categories are kept, and the only inherited attributes are the keys.
3. Remove all m:m Relations: An additional category is inserted, that will substitute the m:m relation by m:1 relations to each of the categories involved in the original relation. The keys of the categories originally related with the m:m relation will compose the key of this inserted category.
4. Check for integrity constraints and add any additional one. [Rishe-92]

---

[4] Convention: the attribute constrained to be the key is named with the suffix -key.

[5] Convention: when a category is constrained to have exactly one key, and the key is composed of several attributes, these attributes are named with the suffix -in-key.

III. EVOLUTION OF THE WESTERN CULTURES DATABASE PROJECT

High Performance Database Research Center

The High Performance Database Research Center (HPDRC), a research division of the Florida International University School of Computer Science, conducts research on Semantic Database Management Systems (Semantic DBMS), making possible the development of new applications based on this methodology.

FIU President's Office

The Florida International University President's assists the President in managing the university and planning major events to promote this learning environment. However, the President often develops research projects to study interesting topics related to the university. Some of those projects are based on determining what makes universities successful learning environments, what common characteristics are shared by top ranking universities, and what are the important aspects of our community, environment, and history. While there is no official research department within the President's office, in practice, there are a few people who are continuously assigned to research projects, and their work is coordinated by the President's Executive Research Assistant.

In the beginning of 1996, President Maidique proposed a research topic based on acquiring greater understanding of Western Culture and of the variables (such as economy, trade, location, religion, etc.) that have played a role in developing culture

throughout time and across nations. This is done through pattern searching and extensive data analysis collected in detail by the research team.

Western Cultures Database application

Recently, HPDRC committed to developing a Western Cultures Database application for the FIU President's office. This application is required to store data about persons considered to be historical figures (according to book references collected by the office), due to their contributions in a specific area (field of work). Moreover, the application should generate reports and graphics to support research assertions about the impact of historical facts in the evolution of western cultures.

The **goal** of the project can be specified as follows: "Ultimately the data and it findings will be made available through the web for public access to anyone interested in researching about the findings of Western Cultures or about the impact of a particular Historical Figure in time". Moreover, the idea of publishing a book about Western Cultures, based on the President's office research, is being considered.

Initial Approach: an Analysis phase

The project started with an Analysis phase, where members of HPDRC and the President's office met to discuss the project and formalize the requirements of the application. The HPDRC team followed the premise of understanding the environment in which the application will be used, to develop a clear idea of what the user expects the system to accomplish.

Design phase

Because of the impact of Semantic modeling, and taking into consideration the theoretical base and the advantages of the Semantic model, the database was designed and modeled using this methodology.

In the Design phase a Semantic schema was generated and revised to make it an accurate representation that captures and documents the specification of information. The Semantic Schema Editor v1.0a, developed at HPDRC was initially used to generate the schema. This tool provides a graphical interface to create a semantic schema, and save its definition in an ".sdl" file. See APPENDIX B (design appendix) for a look to the .sdl file.

Later on, the .sdl this file can be used to generate an ".sql" file (for creating the database with the Oracle Relational tool), and a ".cpp" file (for creating the database with the Semantic Binary Database C++ Interface).

The latest version of the Semantic Schema is presented in **Figure 3.1**, **Figure 3.2**, and **Figure 3.3**.

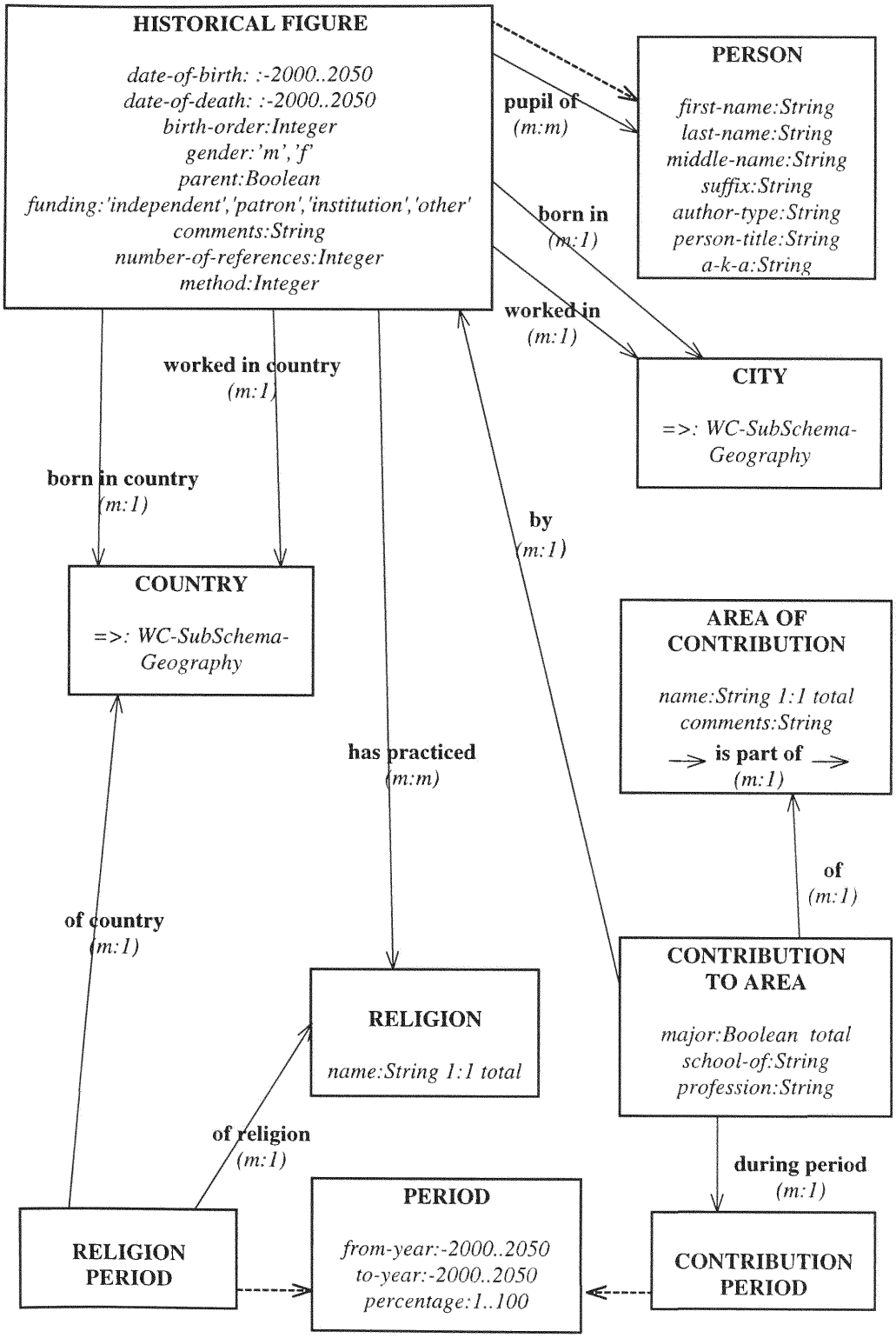


Figure 3.1 Western Culture, Semantic main schema

- ☐ *CITY* — category (See subschema WC-SubSchema-Geography.)
- ☐ *COUNTRY* — category (See subschema WC-SubSchema-Geography.)
- ☐ *PERSON* — category (A record of all persons relevant to this database)
- ☐ *first-name* — attribute of *PERSON*, range: *String* (*m:1*) (The first name of a person)
- ☐ *last-name* — attribute of *PERSON*, range: *String* (*m:1*) (The last name of a person)
- ☐ *middle-name* — attribute of *PERSON*, range: *String* (*m:1*) (The middle name of a person.)
- ☐ *suffix* — attribute of *PERSON*, range: *String* (*m:1*) (The suffix of a name, e.g. Jr in Martin Luther King Jr., II in Elizabeth II.)
- ☐ *author-type* — attribute of *PERSON*, range: *String* (*m:1*) (The type of the person when is an author of a book)
- ☐ *person-title* — attribute of *PERSON*, range: *String* (*m:1*) (The title of a person)
- ☐ *a-k-a* — attribute of *PERSON*, range: *String* (*m:1*) (The name as the person is known as)
- ☐ *CONTRIBUTION-TO-AREA* — category (A record of associations of historical figures with specific areas)
- ☐ *major* — attribute of *CONTRIBUTION-TO-AREA*, range: *Boolean* (*m:1,total*) (This attribute distinguishes a contribution considered a major contribution)
- ☐ *school-of* — attribute of *CONTRIBUTION-TO-AREA*, range: *String* (*m:1*) (The school to where the Historical Figure made the contribution)
- ☐ *profession* — attribute of *CONTRIBUTION-TO-AREA*, range: *String* (*m:1*) (The profession to which the Historical Figure made the contribution)
- ☐ *AREA-OF-CONTRIBUTION* — category (A catalog of areas of contribution)
- ☐ *name* — attribute of *AREA-OF-CONTRIBUTION*, range: *String* (*1:1,total*) (The name of the area of contribution)
- ☐ *comments* — attribute of *AREA-OF-CONTRIBUTION*, range: *String* (*m:1*) (A comment)

- ☐ *RELIGION* — category (A catalog of religions denominations widespread in some countries)

- ☐ *name* — attribute of *RELIGION*, range: *String* (*1:1,total*) (The name of a religion)

- ☐ *RELIGION-PERIOD* — category (A record of religions of particular countries during specific time periods. A country can have several widespread religions at the same time. In addition, countries can change their religion over time.)

- ☐ *PERIOD* — category (A period of time)

- ☐ *from-year* — attribute of *PERIOD*, range: *-2000..2050* (*m:1*) (The year of period beginning.)

- ☐ *to-year* — attribute of *PERIOD*, range: *-2000..2050* (*m:1*) (The year of period end.)

- ☐ *percentage* — attribute of *PERIOD*, range: *1..100* (*m:1*) (An estimate of percentage of the population of the country which practiced the religion or percentage of people who contributed on a specific area during a particular period)

- ☐ *CONTRIBUTION-PERIOD* — category (A record of contribution of a particular historical person to specific area during time periods.)

- ☐ *HISTORICAL-FIGURE* — category (Historical figures are persons who are subjects of this study.)

- ☐ *date-of-birth* — attribute of *HISTORICAL-FIGURE*, range: *-2000..2050* (*m:1*) (The date of birth, if known)

- ☐ *date-of-death* — attribute of *HISTORICAL-FIGURE*, range: *-2000..2050* (*m:1*) (The date of death, if known)

- ☐ *birth-order* — attribute of *HISTORICAL-FIGURE*, range: *Integer* (*m:1*) (A generation which the historical figure belongs to, they are ordinals such as 1,2,3,4,5.., they represent whether the person is a first born, second born)

- ☐ *gender* — attribute of *HISTORICAL-FIGURE*, range: *'m','f'* (*m:1*) (The gender of the person , i.e. m:male, f:female)

- ☐ *parent* — attribute of *HISTORICAL-FIGURE*, range: *Boolean* (*m:1*) (The parent's occupation. If it is true it means that the person's father did the same work as the person)

- ☐ *funding* — attribute of *HISTORICAL-FIGURE*, range:

'independent', 'patron', 'institution', 'other' (*m:1*) (The type of funding the person received)

- ☐ *comments* — attribute of *HISTORICAL-FIGURE*, range: *String* (*m:1*) (Comment for Historical Figure. May be used also to indicate that the country of birth is different from country of work)

- ☐ *number-of-references* — attribute of *HISTORICAL-FIGURE*, range: *Integer* (*m:1*) (The number of references of that Historical Figure in the literature)

- ☐ *method* — attribute of *HISTORICAL-FIGURE*, range: *Integer* (*m:1*) (The method's number used to collect the information about a Historical Figure)

- ☐ *of* — relation from *CONTRIBUTION-TO-AREA* to *AREA-OF-CONTRIBUTION* (*m:1*) (The area to which a contribution was made)

- ☐ *by* — relation from *CONTRIBUTION-TO-AREA* to *HISTORICAL-FIGURE* (*m:1*) (The historical figure by whom the contribution was made)

- ☐ *during-period* — relation from *CONTRIBUTION-TO-AREA* to *CONTRIBUTION-PERIOD* (*m:1*) (The period during which a contribution was made)

- ☐ *is-part-of* — relation from *AREA-OF-CONTRIBUTION* to *AREA-OF-CONTRIBUTION* (*m:1*) (The area of which this is a subarea)

- ☐ *of-religion* — relation from *RELIGION-PERIOD* to *RELIGION* (*m:1*) (The subject religion during a particular period)

- ☐ *of-country* — relation from *RELIGION-PERIOD* to *COUNTRY* (*m:1*) (The subject country)

- ☐ *has-practiced* — relation from *HISTORICAL-FIGURE* to *RELIGION* (*m:m*) (The religion practiced by a historical figure. A person may have practiced several religions)

- ☐ *worked-in* — relation from *HISTORICAL-FIGURE* to *CITY* (*m:1*) (The modern country within which modern borders the person was working)

- ☐ *born-in* — relation from *HISTORICAL-FIGURE* to *CITY* (*m:1*) (The modern country within which modern borders the person was born)

- ☐ *born-in-country* — relation from *HISTORICAL-FIGURE* to *COUNTRY* (*m:1*) (The country where the historical figure was born)

☐ *worked-in-country* — relation from *HISTORICAL-FIGURE* to *COUNTRY* (*m:1*) (The country were the Historical Figure worked in)

☐ *pupil-of* — relation from *HISTORICAL-FIGURE* to *PERSON* (*m:m*) (The teacher of that Historical Figure)

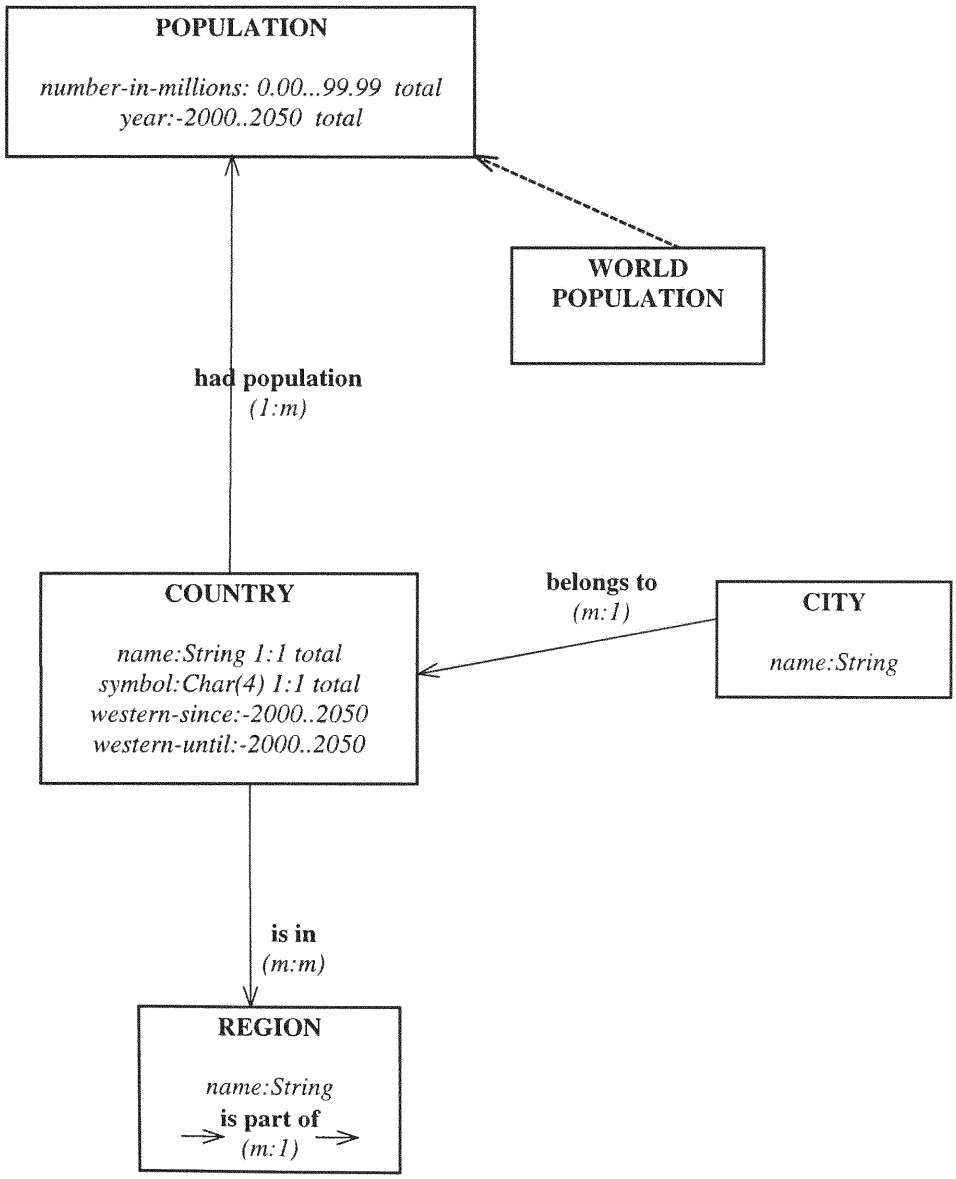


Figure 3.2 Western Culture - Geography sub-schema

☐ *REGION* — category (A region, sub-region, continent, or any other geographical area

which may be of interest)

- ☐ *name* — attribute of *REGION*, range: *String* (*m:1*) (A name of the Region)

- ☐ *WORLD-POPULATION* — category

- ☐ *POPULATION* — category (A record of censuses or estimates of population of countries and the world)

- ☐ *number-in-millions* — attribute of *POPULATION*, range: *0.00...99.99* (*m:1,total*) (The population)

- ☐ *year* — attribute of *POPULATION*, range: *-2000..2050* (*m:1,total*) (The year of census or estimate of the population)

- ☐ *CITY* — category (The record of all cities relevant to this study)

- ☐ *name* — attribute of *CITY*, range: *String* (*m:1*)

- ☐ *COUNTRY* — category (Table of countries relevant to this study)

- ☐ *name* — attribute of *COUNTRY*, range: *String* (*1:1,total*) (The modern name of the Country)

- ☐ *symbol* — attribute of *COUNTRY*, range: *Char(4)* (*1:1,total*) (The international 4-letter symbol of the country, e.g. US,RU,AU,CA,USVI)

- ☐ *western-since* — attribute of *COUNTRY*, range: *-2000..2050* (*m:1*) (The beginning year in which the country was considered part of Western Civilization)

- ☐ *western-until* — attribute of *COUNTRY*, range: *-2000..2050* (*m:1*) (The beginning year in which the country was considered part of Western Civilization)

- ☐ *is-part-of* — relation from *REGION* to *REGION* (*m:1*)

- ☐ *belongs-to* — relation from *CITY* to *COUNTRY* (*m:1*)

- ☐ *had-population* — relation from *COUNTRY* to *POPULATION* (*1:m*) (Population record of the Country)

- ☐ *is-in* — relation from *COUNTRY* to *REGION* (*m:m*)

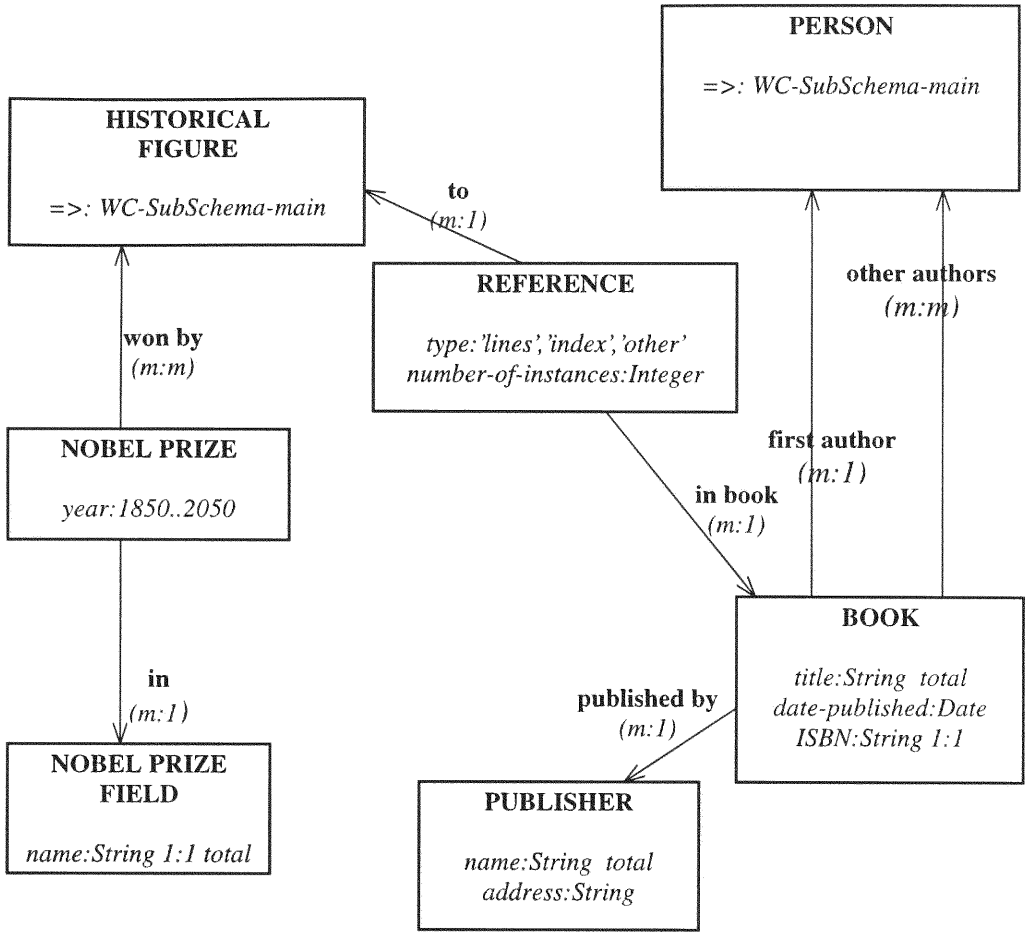


Figure 3.3 Western Culture- Prizes and Citations semantic sub-schema

- ☐ *PERSON* — category (See subschema WC-SubSchema-main.)
- ☐ *HISTORICAL-FIGURE* — category (See subschema WC-SubSchema-main.)
- ☐ *NOBEL-PRIZE* — category (A catalog of Nobel Prizes)
- ☐ *year* — attribute of *NOBEL-PRIZE*, range: *1850..2050* (*m:1*) (The year in which a Nobel Prize was awarded to a particular person)
- ☐ *NOBEL-PRIZE-FIELD* — category (A catalog of fields in which Nobel Prizes are awarded)

- ☐ *name* — attribute of *NOBEL-PRIZE-FIELD*, range: *String* (*1:1,total*) (The name of the field of the Nobel Prize)

- ☐ *REFERENCE* — category (A catalog of references to persons in books)

- ☐ *type* — attribute of *REFERENCE*, range: *'lines', 'index', 'other'* (*m:1*) (The type of reference made. e.g. by counting lines, by looking in the index)

- ☐ *number-of-instances* — attribute of *REFERENCE*, range: *Integer* (*m:1*) (The number of references to a particular historical figure found in the book)

- ☐ *PUBLISHER* — category (A catalog of publishers)

- ☐ *name* — attribute of *PUBLISHER*, range: *String* (*m:1,total*) (The name of the publisher)

- ☐ *address* — attribute of *PUBLISHER*, range: *String* (*m:1*) (The address of the publisher)

- ☐ *BOOK* — category (A catalog of books in which references were located)

- ☐ *title* — attribute of *BOOK*, range: *String* (*m:1,total*) (The title of the book)

- ☐ *date-published* — attribute of *BOOK*, range: *Date* (*m:1*) (The date of publication, if known)

- ☐ *ISBN* — attribute of *BOOK*, range: *String* (*1:1*) (The International Standard Book Number, if known)

- ☐ *in* — relation from *NOBEL-PRIZE* to *NOBEL-PRIZE-FIELD* (*m:1*) (The field in which the Nobel Prize was won)

- ☐ *won-by* — relation from *NOBEL-PRIZE* to *HISTORICAL-FIGURE* (*m:m*) (The historical figure who won the Nobel Prize)

- ☐ *to* — relation from *REFERENCE* to *HISTORICAL-FIGURE* (*m:1*) (The historical figure who is referenced)

- ☐ *in-book* — relation from *REFERENCE* to *BOOK* (*m:1*) (The book in which the reference was found.)

- ☐ *published-by* — relation from *BOOK* to *PUBLISHER* (*m:1*) (The publisher of the book, if known)

- ☐ *first-author* — relation from *BOOK* to *PERSON* (*m:1*) (The main person who wrote the book)
- ☐ *other-authors* — relation from *BOOK* to *PERSON* (*m:m*) (The co-authors of the book, other than the first author.)

Is important to mention that in this Schema, the BOOK category will have to be converted in two categories: BOOK and JOURNAL ARTICLE. This should be done to also reflect future data about journals and not only about books. See APPENDIX B (design appendix) for a proposed Prizes and Citations semantic sub-schema.

Data Collection

The President's office collected the data about Historical Figures categorizing them as: Scientists, Artists and Philosophers. More data will be collected in the fields of Music, Literature and other areas because this is the most efficient (small and simple) way of grouping persons in major areas according to their contribution to western culture.

The data collection can be subjective and long. In order to create criteria for limiting the number of Historical Figures included in the study, six paramount books where referenced to limit the data to a reasonable size [6]. Following this approach, the President's office reunited data small enough to work without leaving out anyone important, but big enough to have information and draw valid conclusions from the research.

The data was collected in Excel files, following the format presented in **Figure 3.4**

[6] The referenced books are: 1.- A Biographical Dictionary of Artists, 2.- Phaidon Encyclopedia of Art and Artists, 3.- The New Everyman Dictionary of Music, 4.- The Oxford Companion to Philosophy, 5.-Honderich A&E Biography.

Last Name	**First Name**	**Middle Name**	**Suffix**	**Person's Title**	**A.K.A.**	**Gender**	**Number of References**
Birth Year	**Death Year**	**Birth City**	**Birth Country**	**Nobel Prize**	**Year for the Nobel Prize**	**Second Nobel Prize**	**Year for the Second Nobel Prize**
Major Contribution	**School of** (for Major Contribution)	**Profession** (for Major Contribution)	**Secondary Contribution**	**School of** (for Secondary Contribution)	**Profession** (for Secondary Contribution)		
Comments	**Method**						

Figure 3.4 Format used for Data Collection

In **Figure 3.4** the data about a Historical Figure is presented providing information about his complete name, a possible title and nobility, an alias (A.K.A), it's Gender and the number of references found about that person in the literature. Also this files contain data about the Historical Figures place and date of birth, date of death, acquisition of Nobel Prizes and contribution to Areas (in a particular school, for a particular profession).
In the APPENDIX C (data appendix) are included snapshots of the Art, Science and Philosophy files provided by the President's office on 6/12/98.

Implementation phase

The initial approach for the Implementation phase of the project was to consider a Relational methodology, making necessary then a model conversion, to obtain the design of relational tables. Following the model conversion methodology (explained in the Theoretical Background, chapter II), from the Semantic model, a Relational model was

obtained.[7] See APPENDIX D (relational appendix) for the result of the model conversion (Relational Schema of the application).

The Semantic approach was considered also for the Implementation phase because of the theoretical advantages of the Semantic modeling. All the capabilities of a Relational approach can be implemented using the Semantic/Object Oriented tools available at HPDRC (Sem-ODB and Web-RG), and it requires less effort and time for development and maintenance. Furthermore implementing the application using the Relational approach, does not permit the user to take advantage of the Internet (lack of an Oracle tool for putting the application on the web).

The Western Cultures project started then from a common Design, and using the two implementation approaches, continued with them in parallel, reaching a point where the Relational one is left aside, and the Semantic one goes beyond in facilitating the "Western Cultures Semantic Database" application.

Relational Implementation approach

1. Database Creation:

The first step on the implementation of the project, using a Relational approach was to create the database. The database was created based on the conversion of the design (from Semantic to Relational), explained above. See APPENDIX D (relational appendix) for the script file (.sql file) to create the database.

[7] The Union+Events option was used for the convertion.

2. Database Population:

The second step of the Relational approach was to load the database, using the text files (Excel files) provided by the user. Some of the tables were successfully loaded, but this process is still pending (not finished). See APPENDIX D (relational appendix) for the script files (.ldr files) to load some tables of the database.

3. Resulting Application: Data Entry Forms:

The Data entry interface, for the Relational approach was implemented using Oracle forms (Oracle 7.0 for Windows NT) to generate a user front end for data update. The possibility for using Oracle graphics and Oracle reports, to cover requirements about data retrieval and presentation was discussed. The following figures show the Oracle Forms.

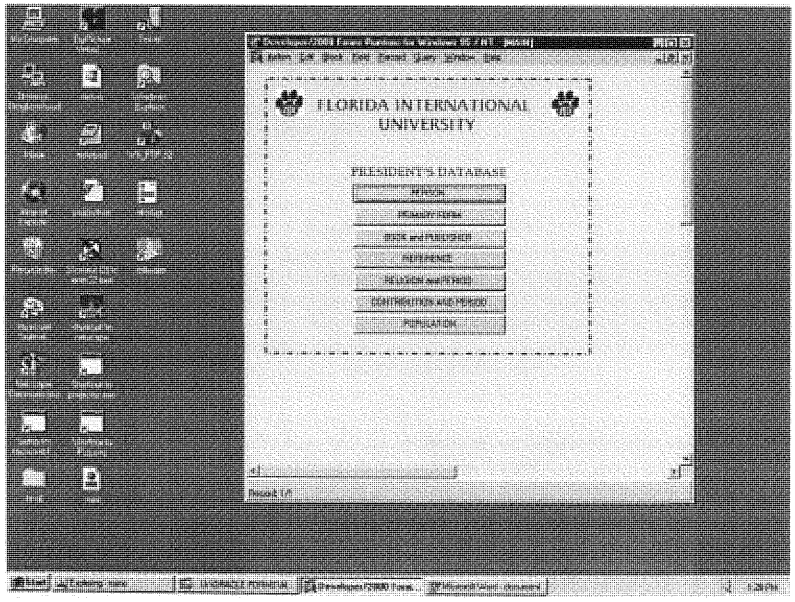


Figure 3.5 Main Form – Oracle Forms

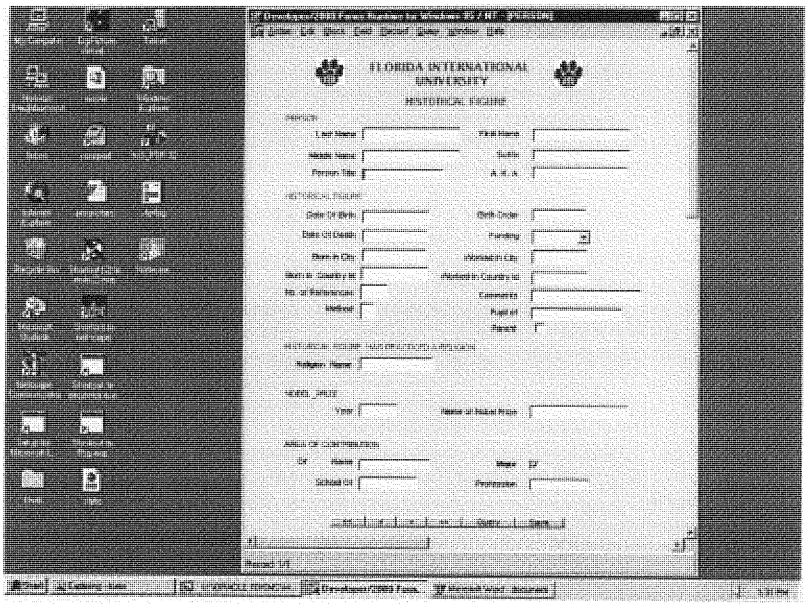


Figure 3.6 Person Form – Oracle Forms

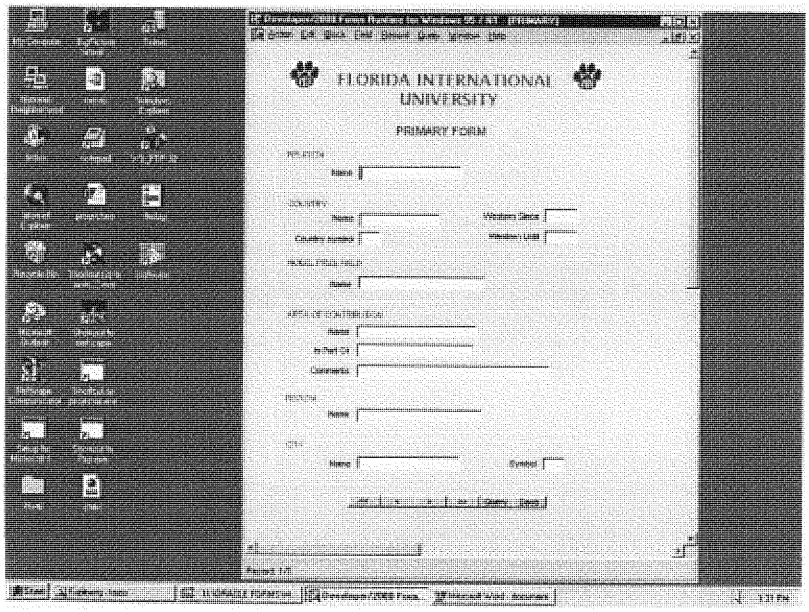


Figure 3.7 Primary Form – Oracle Forms

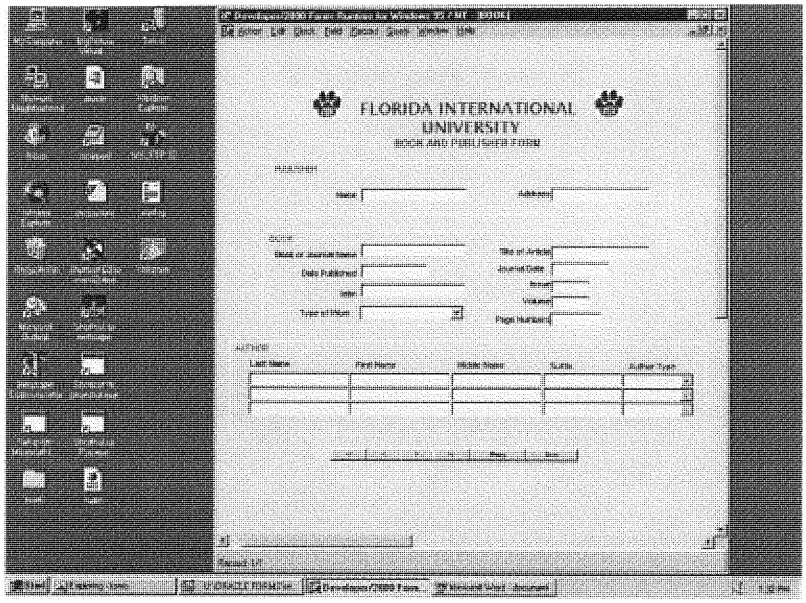


Figure 3.7 Book and Publisher Form – Oracle Forms

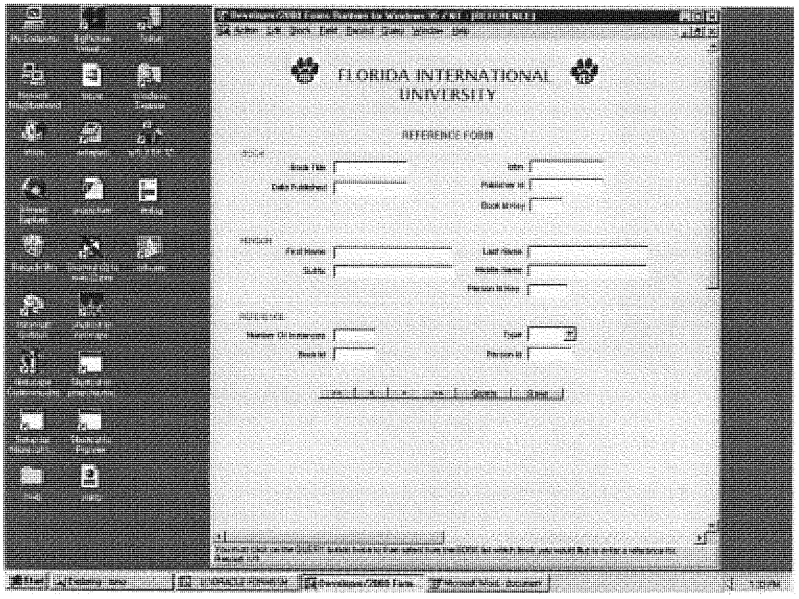


Figure 3.8 Reference Form – Oracle Forms

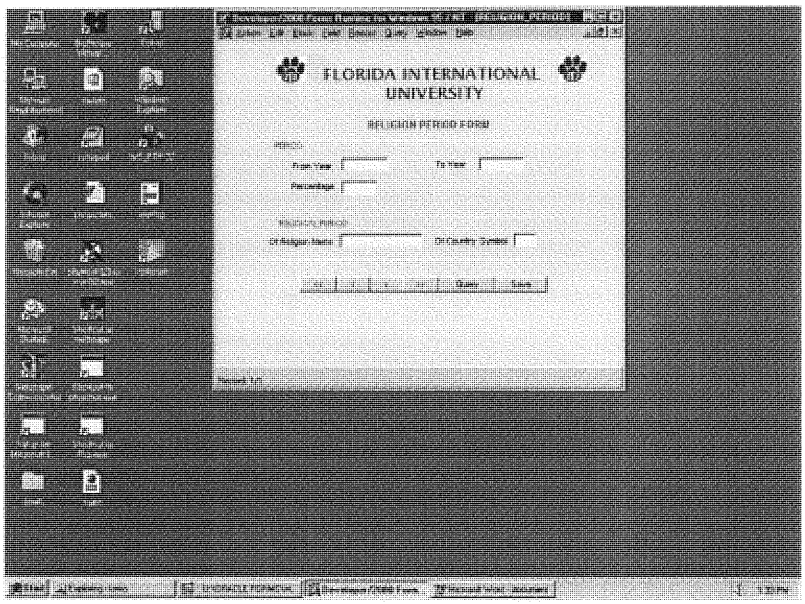


Figure 3.9 Religion Period Form – Oracle Forms

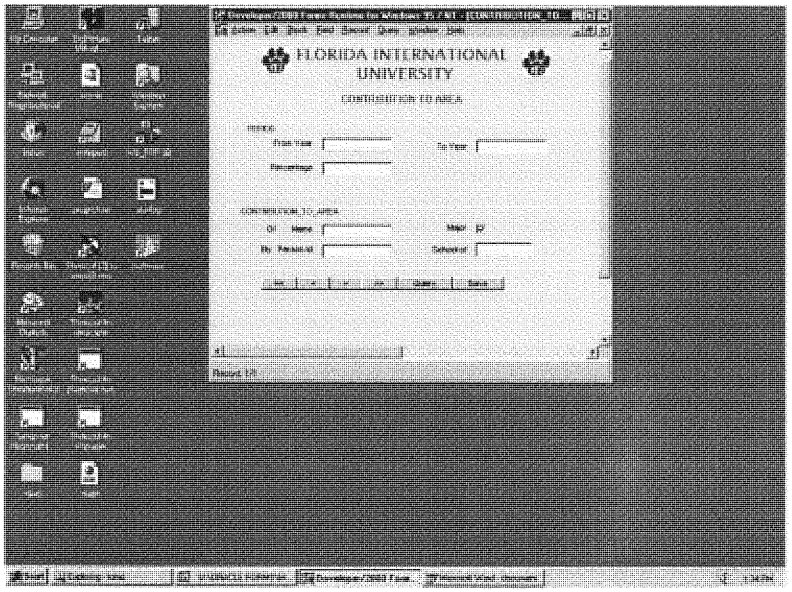


Figure 3.10 Contribution to Area Form – Oracle Forms

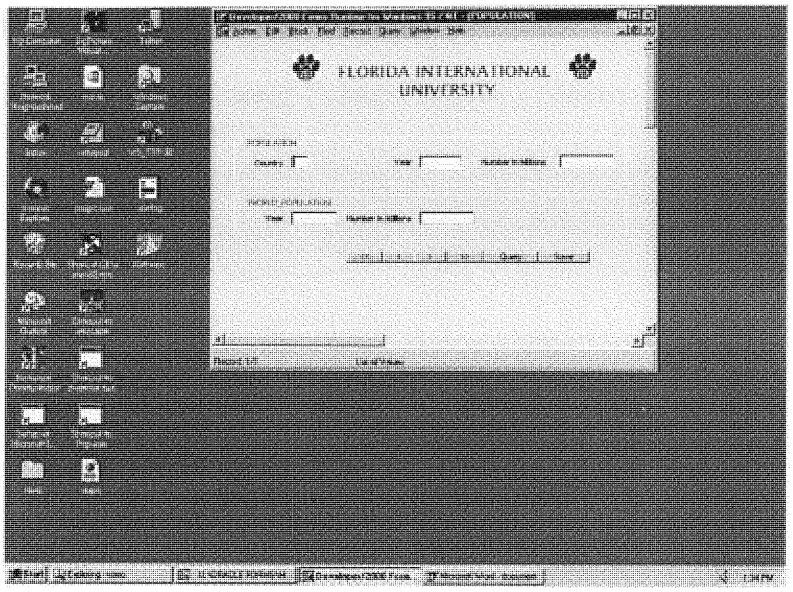


Figure 3.11 Population Form – Oracle Forms

Semantic Implementation (Sem-ODB) approach

The implementation of the "Western Cultures Semantic Database" included the following steps:

1. Database Creation:

The first step consisted of building the structure of the database: categories, attributes for each category, and the relations between categories.

The Semantic Object-oriented database Management Systems (Sem-ODB) interface (a C++ interface developed by HPDRC) is a tool for developing and managing database applications; the "Western Cultures Semantic Database" is application that uses a Sem-ODB as a tool for building and loading the database.

The main routine for creating the database is shown below and consists of three lines of C++ code.

```
TDataBase * DB = CreateDataBase("president.DB");
SchEd_CreateDataBase ("president.DB", DB);
CloseDataBase(DB);
```

The first line creates the database and names it. The second line calls a function CreateDataBase, which takes as a parameter the database, and creates the schema (categories, attributes, and relations) for it, returning the database to be closed in the third line.

The function CreateDataBase consists of the following sections.

```
theDatabase -> Transaction_Begin ( );
        // Creating New types
        NewIntegerClass ( theDatabase, "-2000..2050", -2000, 2050, "range" ) ;
        ...
        // Creating categories and their attributes
        NewCategory ( theDatabase, "PERSON", "A record of all persons relevant
                        to this database" ) ;
        NewRelation ( theDatabase, "FIRST_NAME", "PERSON", "String",
                        CC_MANY_TO_ONE, "The first name of a person") ;
        ...
        NewCategory ( theDatabase, "HISTORICAL_FIGURE", "Historical
                        figures are persons who are subjects of this study." ) ;
        NewRelation ( theDatabase, "DATE_OF_BIRTH",
                        "HISTORICAL_FIGURE", "-2000..2050",
                        CC_MANY_TO_ONE, "The date of birth, if known") ;
        ...
        // Creating inheritance
        MakeInheritance ( "HISTORICAL_FIGURE", "PERSON", theDatabase ) ;
        ...
        // Creating relations
        NewRelation ( theDatabase, "PUPIL_OF", "HISTORICAL_FIGURE",
                        "PERSON", CC_MANY_TO_MANY, "The theacher of
                        that Historical Figure") ;
        ...
theDatabase -> Transaction_End ( ) ;
```

To create the skeleton of a Database, the function starts a transaction where new data types are created, the categories and their attributes are created, the inheritance are made, and the relations are established. The piece of code included above reflects the creation of the PERSON category and the inclusion of its attribute date-of-birth, whose type is a range that goes from –2000 to 2050 (the negative number representing a BC year and the positive number representing an AD year). Also it reflects that a HISTORICAL FIGURE is a sub-category of PERSON (inheritance), and that a HISTORICAL FIGURE can be the

pupil of a PERSON (relation). More information about the Sem-ODB syntax, can be found in [Sem-ODB-98].

The complete program used to build the database is shown in APPENDIX E (Semantic C++ appendix).

2. Database Population

To load the data provided by the President's office, a group of steps were followed to prepare it for the loader program.

- Save the file as a ".csv" file. A ".csv" file is a text file, where the cells are separated by commas (,).
- Add an extra column at the end of the data columns, to represent the end of line.
- Fill all the empty cells with asterisks (*).

Figure 3.12 shows part of the data for Scientists, after following the previous steps (science.csv file).

```
Abel ,Niels,Henrick,*,9,Finnoy,NO,*,*,Math,*,1802-AD,1829-AD,6,*,m,*,*,*,*,*
Adams ,Walter,Sydney,*,5,Antioch,SY,*,*,Astronomy,*,1876-AD,1956-AD,6,*,m,*,*,*,*,*
Addison ,Thomas,*,*,8,Longbenton,UK,*,*,Physician,*,1793-AD,1860-AD,6,*,m,*,*,*,*,*
Agassiz ,Jean,Louis Rodolphe,*,30,Motier-en-Vuly,CH,*,*,Naturalist,*,1807-AD,1873-AD,6,*,m,*,*,*,*,*
Agricola ,Georgius,*,*,13,Glauchau,GR,*,*,Minerology,Metallurgy,1494-AD,1555-AD,6,*,m,Georg Bauer,*,*,*,*
Alder ,Kurt,*,*,2,Chorzow,PL,Chemistry,1950,Organic-Chemistry,*,1902-AD,1958-AD,6,*,m,*,*,*,*,*
Alpini ,Prospero,*,*,3,Marostica,IT,*,*,Botany,*,1553-AD,1616-AD,6,*,m,Prosper Alpinus,*,*,*,*
Amici ,Giovanni,Battista,*,4,Modena,IT,*,*,Astronomy,Natural-Philosophy,1786-AD,1863-AD,6,*,m,*,*,*,*,*
Ampere ,Andre,Marie,*,26,Lyon,FR,*,*,Physics,*,1775-AD,1836-AD,6,*,m,*,*,*,*,*
Andrews ,Thomas,*,*,7,Belfast,UK,*,*,Physics,Chemistry,1813-AD,1885-AD,6,*,m,*,*,*,*,*
Angstrom ,Anders,Jonas,*,4,Lodgo,SE,*,*,Physics,*,1814-AD,1874-AD,6,*,m,*,*,*,*,*
Appleton ,Edwards,Victor,Sir,6,Bradford,UK,Physics,1947,Physics,*,1892-AD,1965-AD,6,*,m,*,*,*,*,*
Appolonius of Perga,*,*,*,13,Alexandria,EG,*,*,Mathematics,*,262-BC,190-BC,6,*,m,Great Geometer,*,*,*,*
```

Figure 3.12 Part of the Scientists data after preparation for loading.

Having the data after the preparation, the loader program is able to extract tokens (recognizing them by their comma separation), saving each token in a linear array. Then each cell of the array corresponds to an attribute or relation for the Historical Figure, and its content will populate the database. This is done for each line of the file, which correspond to a new Historical Figure.

When the loader program gets a token with an "*", it does not save it into the array, leaving it's correspondent cell empty, meaning that the Historical Figure does not have data for that particular field. Notice that the end of line will be always like ",*" and the token will have an asterisk value that will be ignored when filling up the array (and of course when loading the Historical Figure in the database).

More in detail, the C++ program for populating the database includes a main module as follows.

```
TDataBase* DB = OpenDataBase("president.DB");
printf("BEFORE LOADING DATA\n");
LoadGeneralTables(DB);
LoadScienceData(DB);
LoadArtData(DB);
LoadPhiloData(DB);
CloseDataBase(DB);
printf(" DONE ...\n");
```

Where the first line opens the database, then the following lines call functions to load the General Tables (Country and Nobel Prize Field categories) and load the data for Scientists, Artists and Philosophers. Finally the database is closed.

More in detail, the loading steps for the Scientists are presented below.

```
void
```

```
LoadScienceData(TDataBase* DB)
{
  fp = fopen ("science.csv", "r");

  // For all the lines in the file (for all the Historical Figures)
  {
    // filling the array with the extracted tokens
    ...
    // loading the persons (historical figure) data
     do
     {
     DB -> Transaction_Begin();
          Var PERSON = DB->NewAbstract( "PERSON" );
          PERSON.Relate("PERSON::LAST_NAME", V[0] );
          PERSON.Relate("PERSON::FIRST_NAME", V[1] );
          ...
          // Get the Person who will be a Historical Figure
          SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB),V[0] );
          SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
          SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
          SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
          SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
          SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
          SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;
          // Relate the Historical Figure to the data
          do {
                    Var HISTORICAL_FIGURE( R );
                    if (HISTORICAL_FIGURE.IsNull() )
                      break;
                    HISTORICAL_FIGURE.Categorize( "HISTORICAL_FIGURE" );
                    HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::REFERENCE_NUMBER",
                                                    atoi(V[4]) );
                    ...
                    R.Next();
          } while (TRUE);
  } while (DB->Transaction_End() == ec_concurrency);
  // loading the City data and relating it to HF and Country
  do
  {
  DB -> Transaction_Begin();
  // Get the Person who has the name
  ...
  // Get the Country who has the symbol
  ...
  // If there is information about the city of birth
  {
          // See if the City is already in the Database
          // If  the City is NOT in the database,
          {
                    // Create the City object
                    ...
                    // Relate the Person to the City where he was born
                    ...
```

```
                    // Relate the City to the Country symbol
                    ...
            }
    }else {
            // Relate the Person to the Country where he was born
            ...
            }
    } while (DB->Transaction_End() == ec_concurrency);

// loading the Contribution to Area data and relating it to HF and Area of Contribution
// major Contribution to Area
// If there is information about the major Contribution to Area
    {
     do
     {
     DB -> Transaction_Begin();
     // See if this area of contribution is in the database
     // If  the area of contribution is NOT in the database,
            {
                    // Create the Area of Contribution object
                    ...
            }
     // Get the Area of contribution again
     ...
     // Get the Person
     ...
     // Create the Contribution to Area  object
     ...
     // relating the Contribution to Area and the Area of Contribution
     ...
     // relating the Historical Figure with the Contribution to Area
     ...
     } while (DB->Transaction_End() == ec_concurrency);
    }
// secondary Contribution to Area (repreat the same steps as loading the major Contribution)
...

// loading the 1st. Nobel Prize data and relating it to HF and Nobel Prize Field
// If there is information about the 1st. Nobel Prize
    {
     do
     {
     DB -> Transaction_Begin();
     // Get the Person
     ...
     // Get the Nobel Prize Field
     ...
     // Create the Nobel Prize object
     ...
     // relating the Nobel Prize Field and Nobel Prize
     ...
     // relating the Historical Figure with the Nobel Prize
     ...
```

```
    } while (DB->Transaction_End() == ec_concurrency);
  }

// loading the 2nd. Nobel Prize data and relating it to HF and Nobel Prize Field (repreat the same steps as
loading the 1st. Nobel Prize)
...

printf("**************** next person *****************\n");

} // endwhile for each line
} // end of the function
```

In the code there are included command like NewAbstract, Relate, RangeQuery, Var and Categorize to create an object from a category, relate two objects, retrieve a set of objects, select an existing object, and to assign an object to a specific category, respectively. More information about the Sem-ODB syntax can be found in [Sem-ODB-98].

The same idea is used to load the Artists and the Philosophers data. The complete program used to load the database is shown in APPENDIX E (Semantic C++ appendix).

The President's database is created and loaded using the 6th. version of the SDB (binary engine) on NT .The steps to compile and run the creation and loading files are also included in APPENDIX E (Semantic C++ appendix).

3. Resulting Application : "Western Culture Semantic Database application" – Web Interface. A more detailed explanation of the application is included in the following Chapter of this documentation.

IV. THE WESTERN CULTURES SEMANTIC DATABASE APPLICATION - WEB INTERFACE

Tools used for building the application

This application was developed using a tool created by HPDRC called WebRG. WebRG is an application that runs on a Web (HTTP) and allows the creation of dynamic documents. The documents that are created have the simplicity of HTML and the functionality of CGI. WebRG makes it easy to add information stored in databases to static Web pages. It allows to easily develop Web forms and reports for any ODBC-compliant database (most databases as well as Web-ODBC semantic DBMS). This tool merges HTML with database functions to create a dynamic access to databases using designer-defined "macro files" and SQL commands for database management. End-users can then query the database through a series of pre-defined forms and reports provided by the application developer. [WebRG-98]

WebRG made possible to have a user-friendly interface available on the Internet, for the "Western Cultures Semantic Database application".

An additional tool used for the application was JavaScript, to make the application (Web page) interactive, in other words tracking user action and responding with some visible change on the page. In particular JavaScript provided the following solutions for this application [Good-98]:

- Validate the HTML input forms (avoid errors on data input). For example validate that a Person inserted in the database has at least a last name.
- Preprocess the data in the client site before submission to a server. For example, the date of birth of a Historical Figure is presented to the user as a positive number followed by the suffix BC or AD [8], but the year is stored in the database as a negative number for BC or a positive number for AD. What JavaScript does is transform the data from the displaying format to the storing format and vice-versa.
- Preprocess the data before submission to a Java applet. For example, construct a string separated by commas, of values calculated from data retrieved from the database. This string will be the input data for the Y-axis of a statistical chart, generated then by a Java applet.
- Control multiple-frame navigation generated by user choices. For example, display a selection list in a different frame, depending on which option the user selects.
- A friendlier interface for the application.

Moreover, another tool was used for including in the application graphical representation of the data. JavaChart is a pure Java charting solution for web page designers and Java developers of all kinds. This product supports almost any kind of business, financial, or

[8] for a before or after Christ year respectively.

scientific chart (pie charts, bar graphs, line graphs, etc), in applet, class library, and Java Bean format. The applets provide a set of parameters to personalize these charts and adequate them to users requirements [JavaChart-98].

A complete Flow Chart of all the files used for the web interface implementation is shown in APPENDIX F (Semantic web-interface appendix), and because of the big quantity of files used to build the interface, this appendix only includes some of the most relevant code.

How to start the application

To use the web interface, the user must start the server[9]. After this, any user can open a web browser that supports Frames, JavaScript and Java applets. The web address of the application, at the moment, is: n12.cs.fiu.edu/president/index.html.

The "Western Cultures Semantic Database Application" - web interface

The "Western Cultures Semantic Database Application" is composed by three main modules: Input Forms, Reports and Graphs, and Advanced Ad Hoc Retrieving Tools. **Figure 4.1** shows the initial screen for the "Western Cultures Semantic Database application".

[9] The server is a file called ControlServer.exe, located at the moment in C:\temp\DBLocation of N77 machine, lab ECS-263.

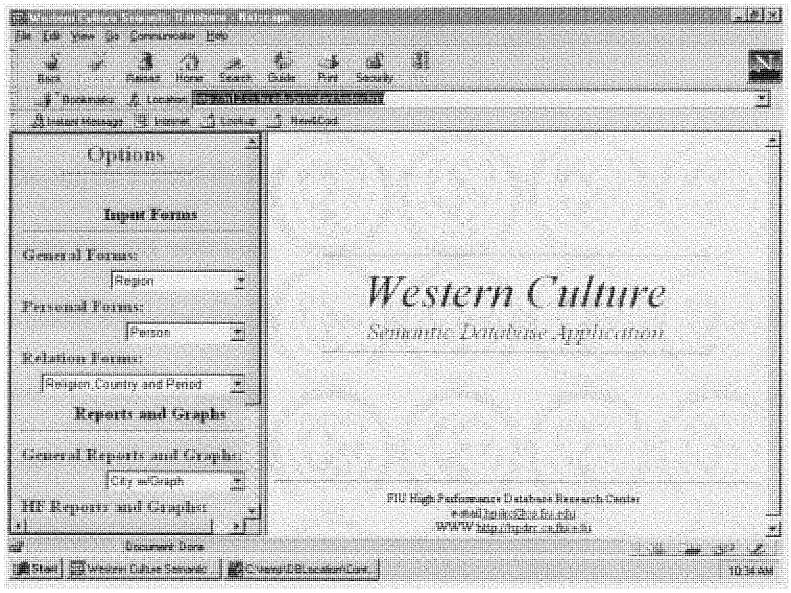


Figure 4.1 "Western Cultures Semantic Database application" – main screen.

The web interface is divided by two frames, the frame Options (left hand side of the screen) and the operation frame (right hand side of the screen). The Options frame is a screen that will remain fixed during the application's use, and in this screen the user can select which option to execute. The operation frame is where the user's selection will be executed.

Each component of the Options frame is explained in detail as follows.

1. Input Forms: Includes HTML input forms for database management: to insert, delete and update (modify) the semantic database content.

   For all the Input Forms of this application, the same design pattern applies: The operation frame is divided in two sub-frames, an upper frame for **Input** a new object, and a lower frame for selecting (from a list of objects) an object that the user wants to

Modify/Remove from the database. Moreover, when the object that the user is inserting (modifying), should be related to another object that is already in the database, the upper frame is divided into two frames, the right-hand frame will be used to display the list where the selection can be done.

The design pattern can be appreciated in any of the snapshots of the application presented below (see the Region Examples 4.1, 4.2 and 4.3 presented below).

More in detail, the Input Forms section contains three sub-sections:

- General Forms: A groups of input forms about general categories in the database that can be loaded independently of any Historical Figure's entry. However, later on the objects of those categories will most likely be related to the Historical Figure objects. The group includes data management for: Region, Country, City, Religion, Nobel Prize Field, Area of Contribution and World Population.

    - Region: This option includes two main sections for data management. The upper section is called "**Input Region**", where the user can insert a new Region in the database, and relate it with another Region that is already in the database (a Region can be part of another Region). To relate the inserted Region, the user can go to the right hand side (white side) of the Input Region, called "**the Region is part of:**" and select from the list of Regions presented. Finally the user just click on the button: Store the Region, and the application gives a confirmation for the data entered in the database. See the Example 4.1 on **Figure 4.2**.

and on **Figure 4.3**, where the South America is inserted and related to America (South America is part of America). **Figure 4.4** is a confirmation screen that shows the new data inserted into the database and give the option of Reload the List of Regions and Insert a new one.

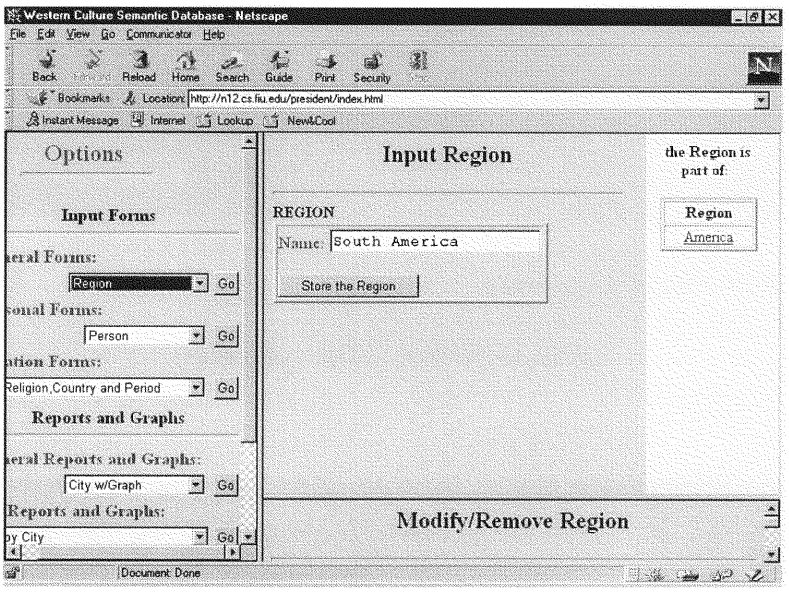


Figure 4.2 Example 4.1 for: Input a new Region where named: South America

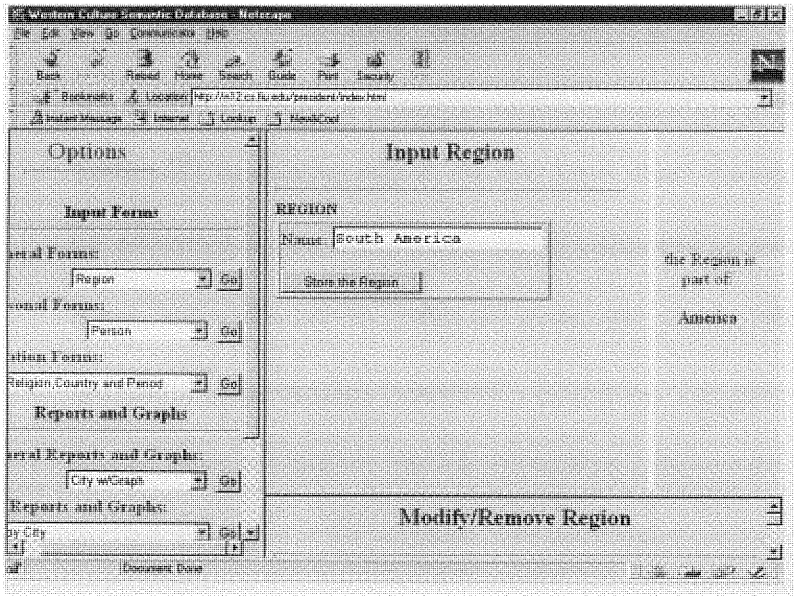


Figure 4.3 …continuation of Example 4.1 for: Input a new Region named: South America, and is part of: America

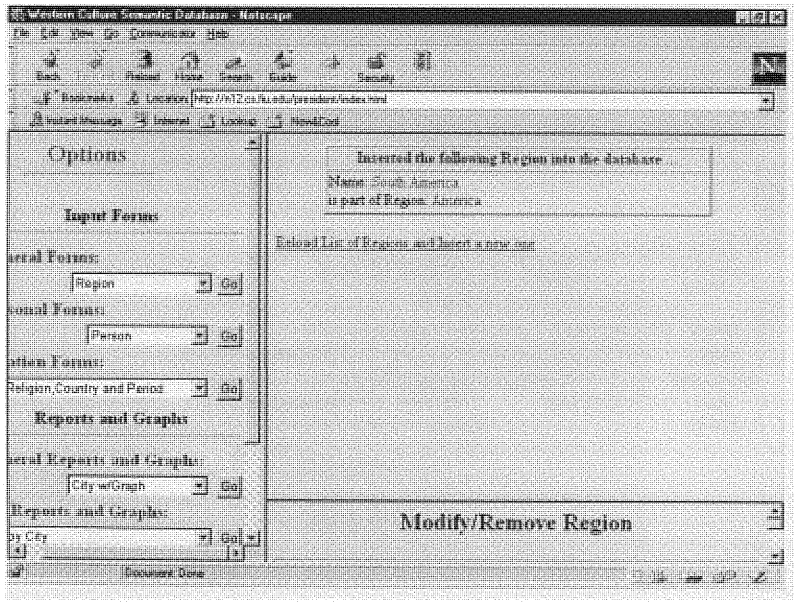


Figure 4.4 Confirmation screen for the previous example (Example 4.1)

The lower section is called "**Modify/Remove Region**", where the user can modify an existing Region or delete it from the database.

To Modify an existing region, the user must select the region from the list presented on the lower frame, and it's data will appear on the upper frame, letting the user to update the name, and select a new region for the relation "**the Region is part of**". Finally the user just click on the button: Store the Region, and the application gives a confirmation for the data modified in the database. See the Example 4.2 on **Figure 4.5**. and on **Figure 4.6**, where South America is updated to South A. and related to Latin America (South A. is part of Latin America). After clicking the Store button, a confirmation screen appears, to show the updated data, and gives the option of Reload the List of Regions and Insert a new one.

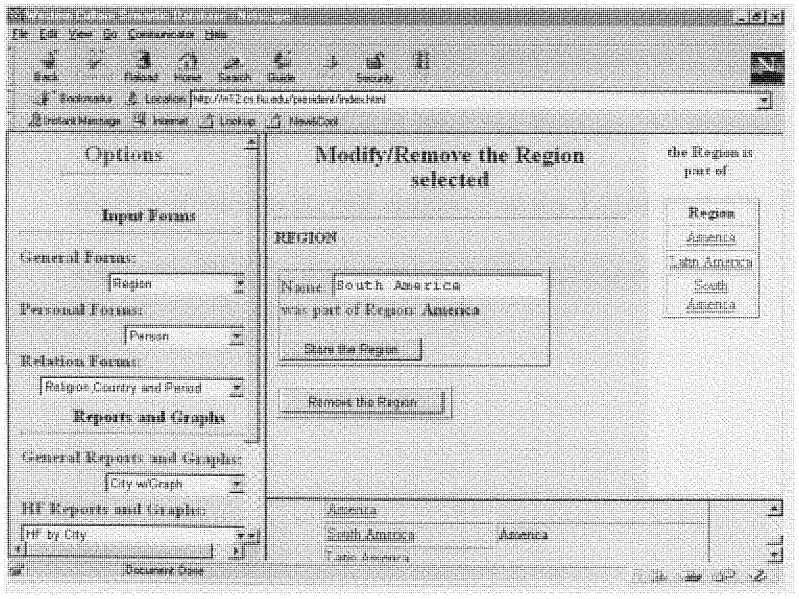


Figure 4.5 Example 4.2 for: Selecting a Region to Modify

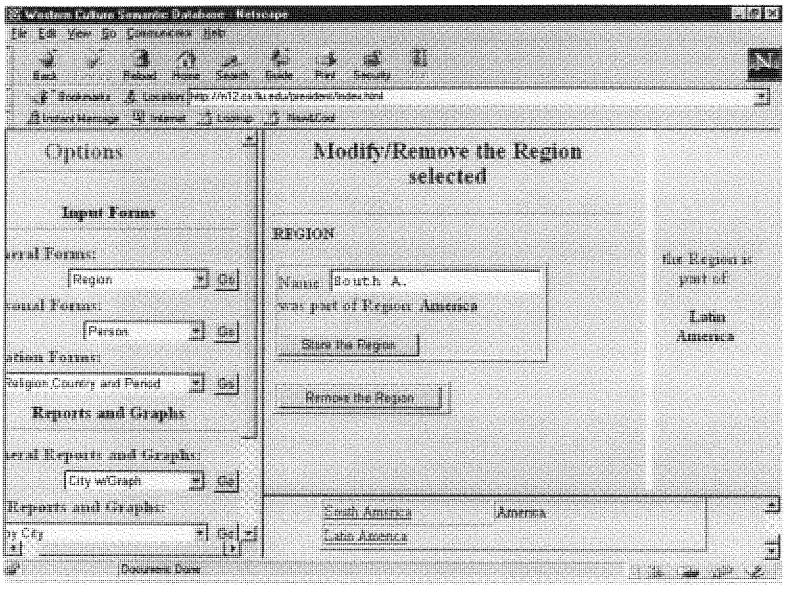


Figure 4.6 ...continuation of Example 4.2 for: Modify a Region from name: South America, and is part of: America, to name: South A., and is part of: Latin America

To Remove an existing region, the user must select the region from the list presented on the lower frame, and it's data will appear on the upper frame, then the user should click the Remove the Region button, and a confirmation screen will appear. See the Example 4.3 on **Figure 4.7** where South A. is Removed from the database.

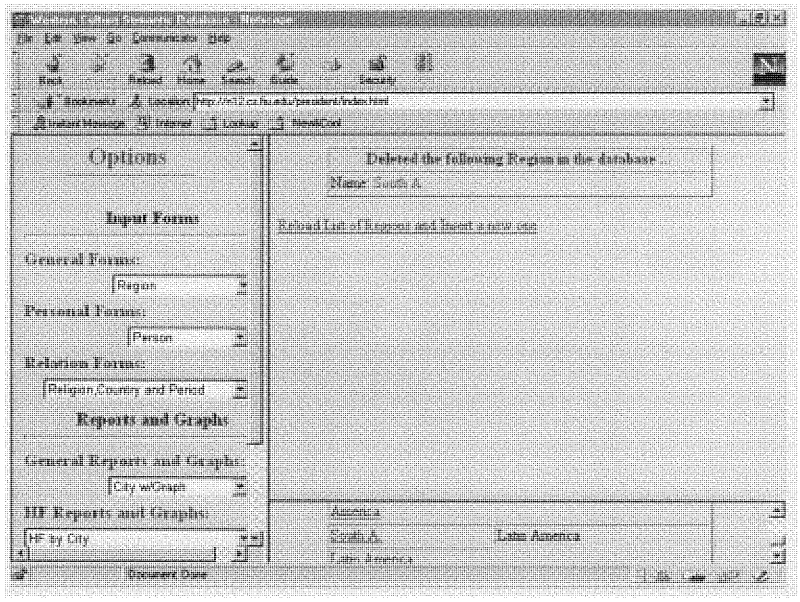


Figure 4.7 Example 4.3 for: Selecting a Region to Remove

- Country: This option includes two main sections for data management. The upper section is called "**Input Country**", where the user can insert a new Country in the database, and relate it with a Region that is already in the database (a Country is in a Region). The lower section is called "**Modify/Remove Country**", where the user can modify an existing Country or delete it from the database. See **Figure 4.8**

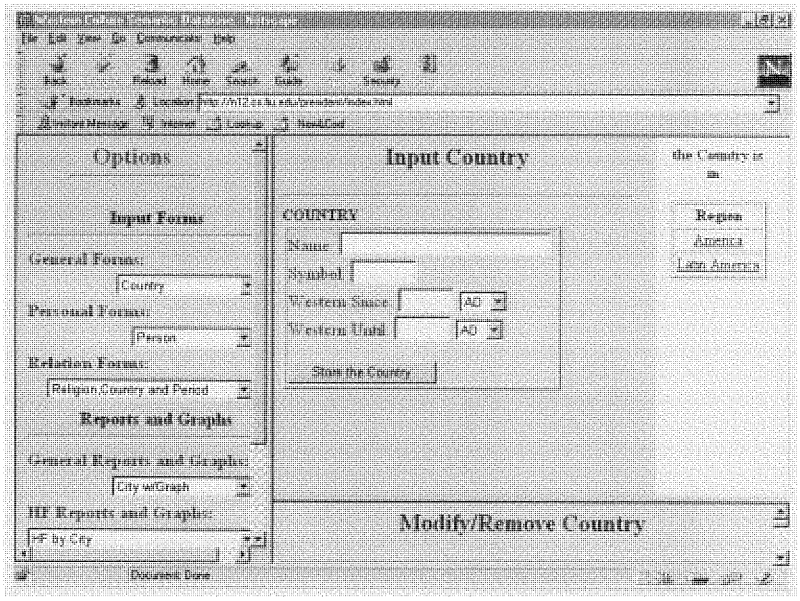


Figure 4.8 Screen for Country General Form

- City: This option includes two main sections for data management. The upper section is called "**Input City**", where the user can insert a new City in the database, and relate it with a Country that is already in the database (a City belongs to a Country). The lower section is called "**Modify/Remove City**", where the user can modify an existing City or delete it from the database. See **Figure 4.9**

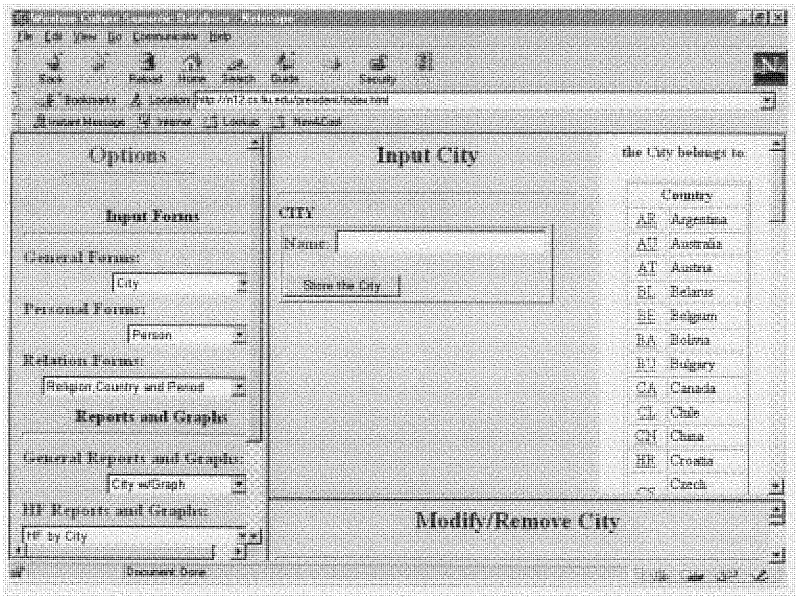


Figure 4.9 Screen for City General Form

- Religion: This option includes two main sections for data management. The upper section is called "**Input Religion**", where the user can insert a new Religion in the database. The lower section is called "**Modify/Remove Religion**", where the user can modify an existing Religion or delete it from the database. See **Figure 4.10**

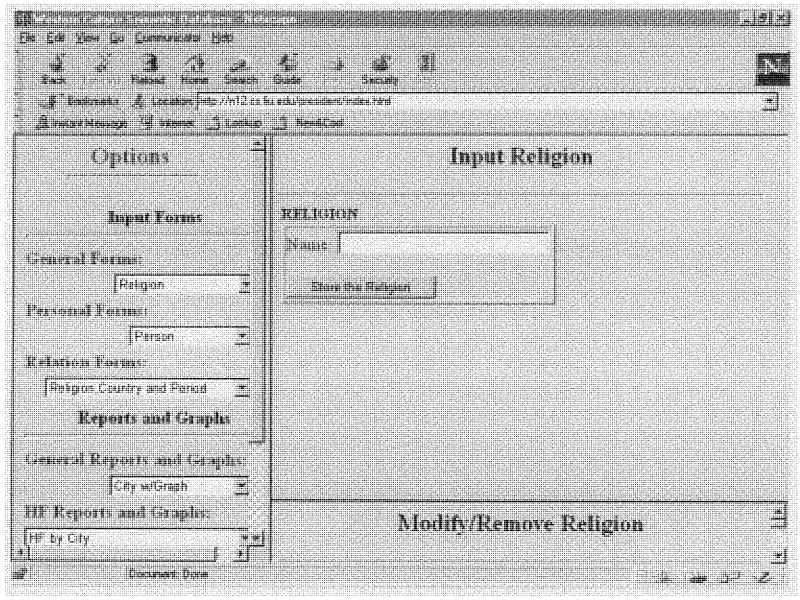


Figure 4.10 Screen for Religion General Form

- Nobel Prize Field: This option includes two main sections for data management. The upper section is called "**Input Nobel Prize Field**", where the user can insert a new Nobel Prize Field in the database. The lower section is called "**Modify/Remove Nobel Prize Field**", where the user can modify an existing Nobel Prize Field or delete it from the database. See **Figure 4.11**

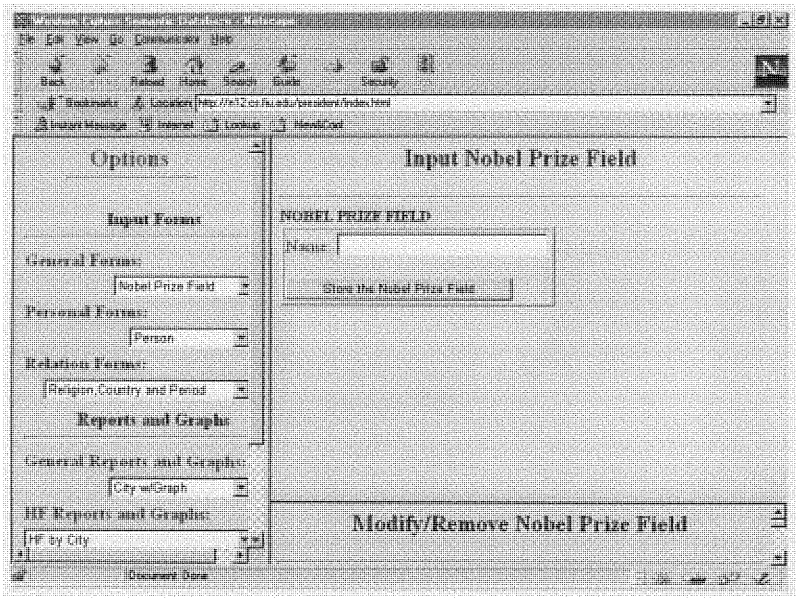


Figure 4.11 Screen for Nobel Prize Field General Form

- Area of Contribution: This option includes two main sections for data management. The upper section is called "**Input Area of Contribution**", where the user can insert a new Area in the database, and relate it with another Area of Contribution, that is already in the database (an Area of Contribution is part of an Area of Contribution). The lower section is called "**Modify/Remove Area of Contribution**", where the user can modify an existing Area or delete it from the database. See **Figure 4.12**

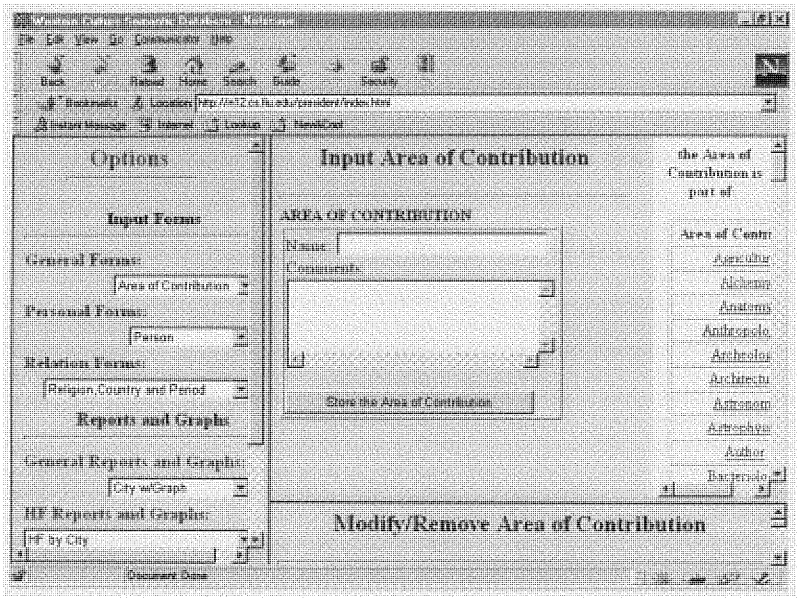


Figure 4.12 Screen for Area of Contribution General Form

- World Population: This option includes two main sections for data management. The upper section is called "**Input World Population**", where the user can insert a year's World Population in the database. The lower section is called "**Modify/Remove World Population**", where the user can modify an existing World Population for a particular year, or delete it from the database. See **Figure 4.13**

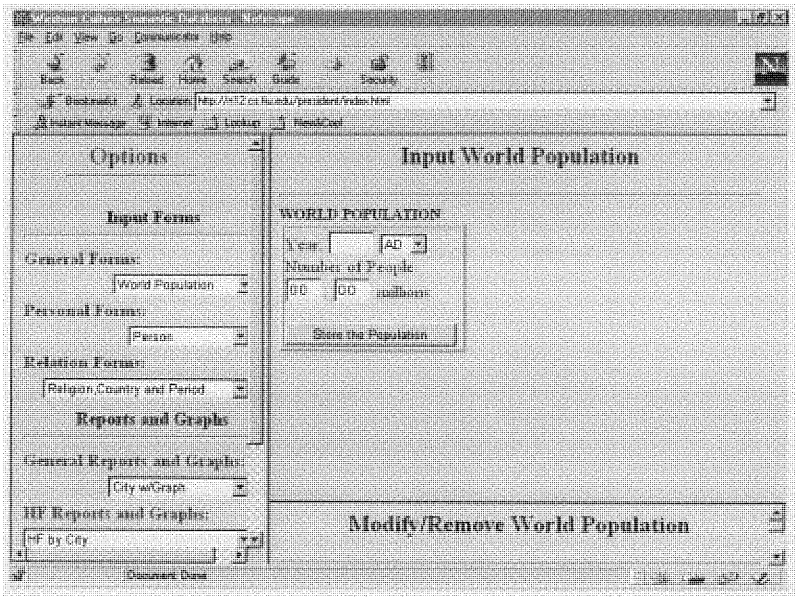


Figure 4.13 Screen for World Population General Form

- Personal Forms: A group of input forms about Person's data entry. The group includes data management for: Person and Historical Figure. There are situations where a Person must be loaded in the database, and that Person is not considered a Historical Figure (i.e., a book author), but, in general, the most important option of this section is the data entry for Historical Figures.

  - Person: This option includes two main sections for data management. The upper section is called "**Input Person**", where the user can insert a new Person in the database. The lower section is called "**Modify/Remove Person**", where the user can modify or delete, an existing Person from the database. See **Figure 4.14**

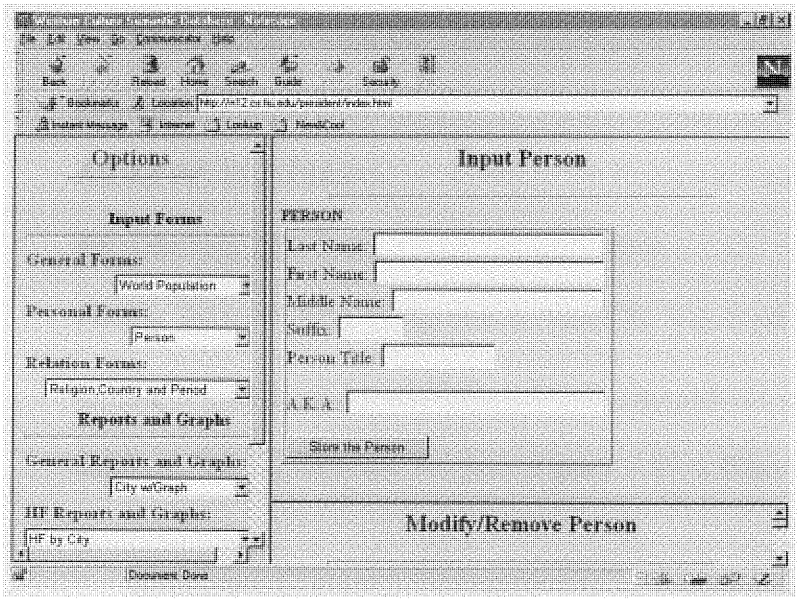


Figure 4.14 Screen for Person in Personal Forms

- Historical Figure: This option is the most important option for data entry, because permits the insertion, modification and deletion of a Historical Figure's data, and his relation to facts already inserted in the database. It includes two main sections for data management.

  The upper section is called "**Input Historical Figure**", where the user can insert data about a new Historical Figure. Also the user can relate the Historical Figure with the place (country or city) where he was born, their teachers, the Religions he has practiced, the Nobel Prized that he has won, and the Contributions he made during his life. To store the relations for inserted Historical Figure, the user just has to click the options for: was born in City or was born in Country, worked

in City or worked in Country, was pupil of Person, has practiced a Religion, has won a Nobel Prize, and made a Contribution.

All the selected objects to be related to the Historical Figure, appear in the right hand side frame called Selected values....

Finally, the user clicks on the button: Store the Historical Figure, and the application gives a confirmation for the data entered in the database. See the Example 4.4 from **Figure 4.15** to **Figure 4.32**.

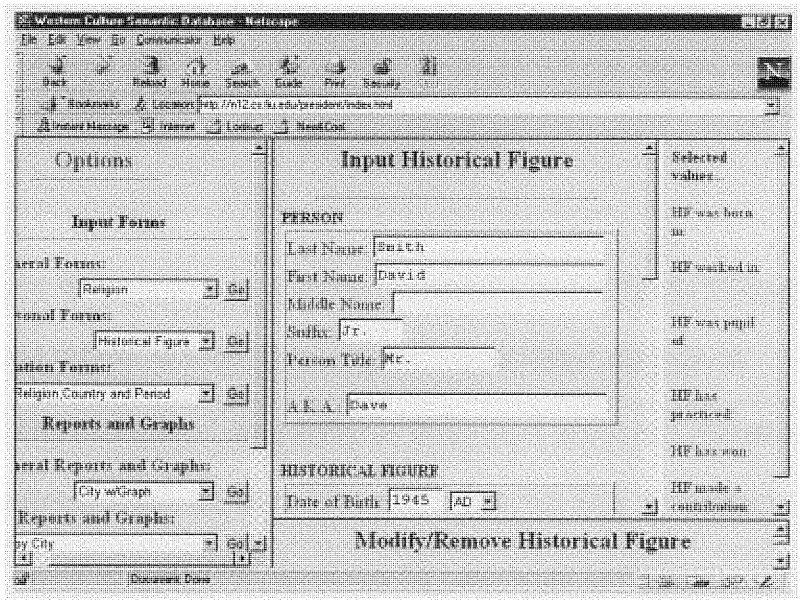


Figure 4.15 Example 4.4 for: Input a new Historical Figure,

Figure 4.15 shows the insertion of the Historical Figure's data entry about his Last Name, First Name, Middle Name, Suffix, Title, and alias (A.K.A).

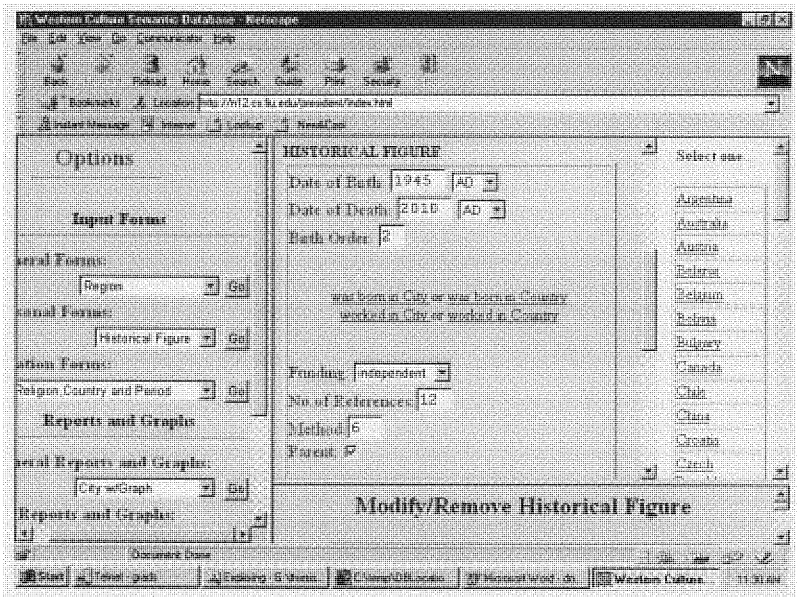


Figure 4.16 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.16 shows a continuation for the Historical Figure's data entry about his Date of Birth, Date of Death, Birth Order, the type of Funding he received, the number of References about him (found in the bibliography), the Method used to collect his data, and a check mark indicating that his parent did the same work as he does. Furthermore this Figure, shows that the user clicked the was born in Country option, and a list of all the countries in the database is presented on the right screen, to help him with the selection of the Country of birth.

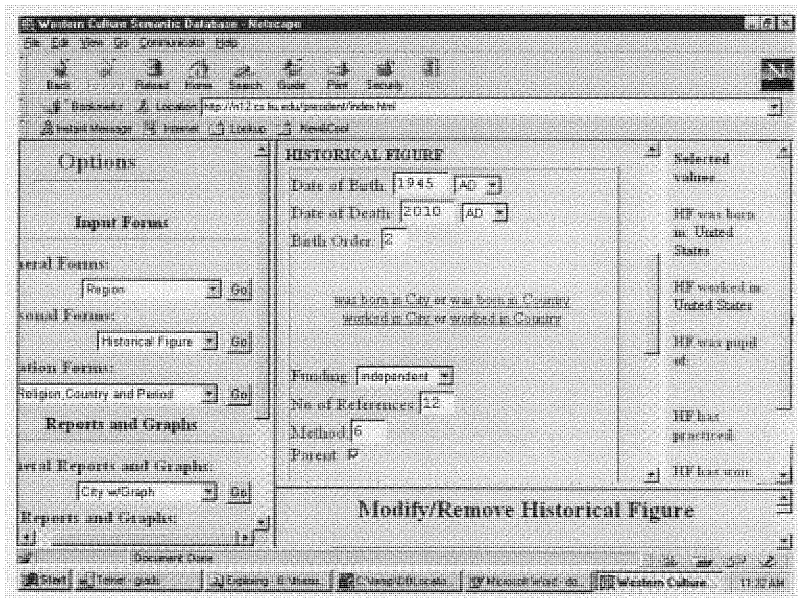


Figure 4.17 … continuation of Example 4.4 for: Input a new Historical Figure

After repeating the same procedure for selecting the Country where the Historical Figure worked, **Figure 4.17** shows the Selected values … on the right-hand frame, at that particular time.

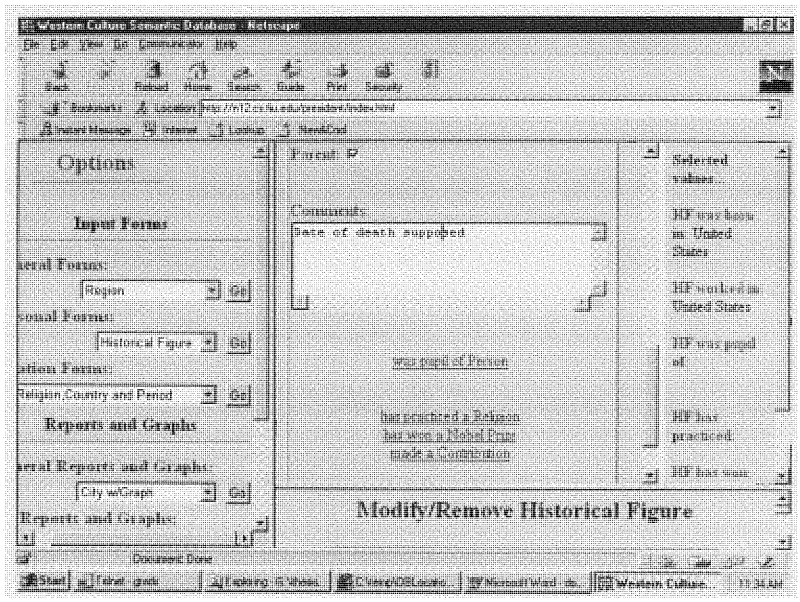


Figure 4.18 … continuation of Example 4.4 for: Input a new Historical Figure

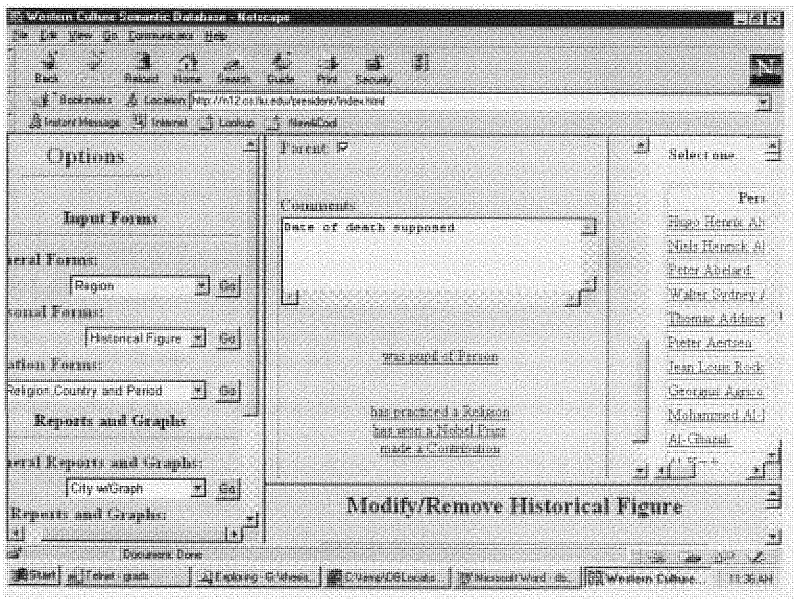


Figure 4.19 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.18 shows a continuation for the Historical Figure's data entry about some necessary Comments.

After clicking the was pupil of Person option, **Figure 4.19** shows the List of all Persons in the database, so the user can pick one of them to be the teacher of the Historical Figure. Then because a Historical Figure can be a pupil of more that one Person (m:m: relation), the user can click again the was pupil of Person option and repeat the selection for another teacher.

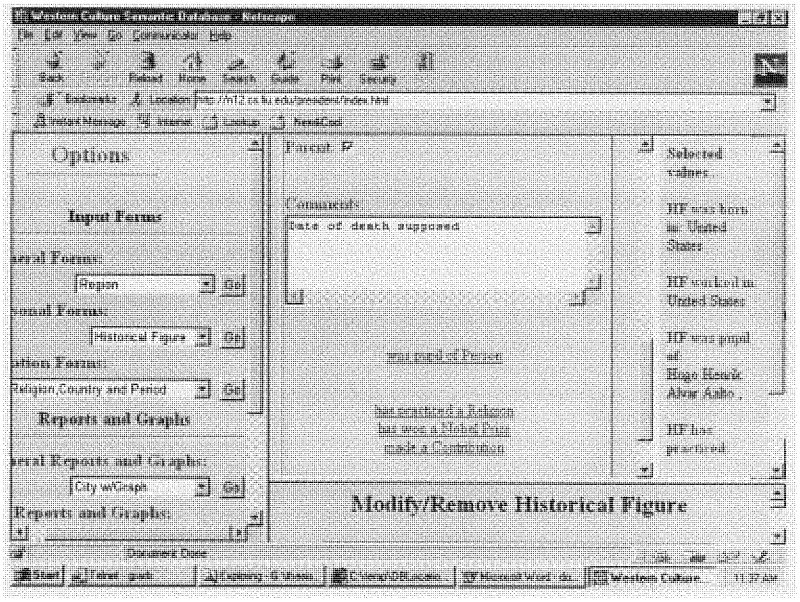


Figure 4.20 ... continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.20 shows that the user selected two Persons to be teachers of the Historical Figure.

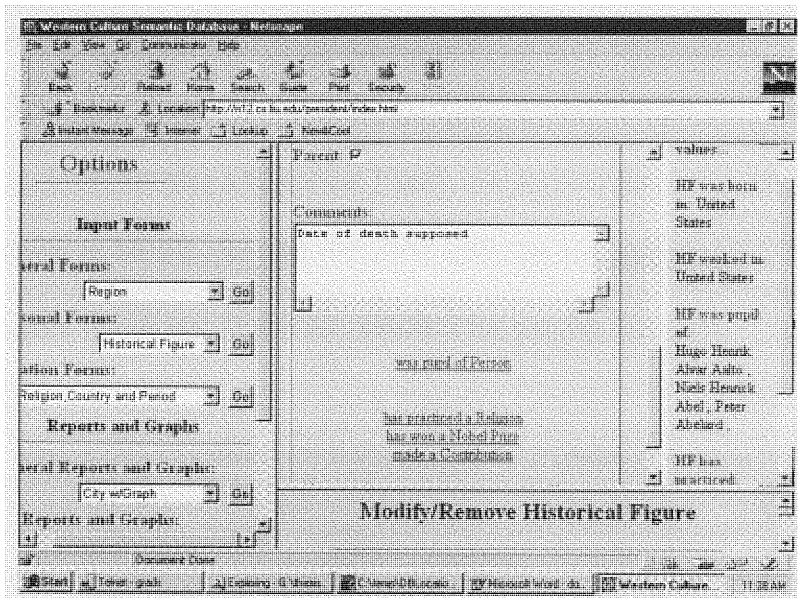


Figure 4.21 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.21 shows that the user then selected another Person.

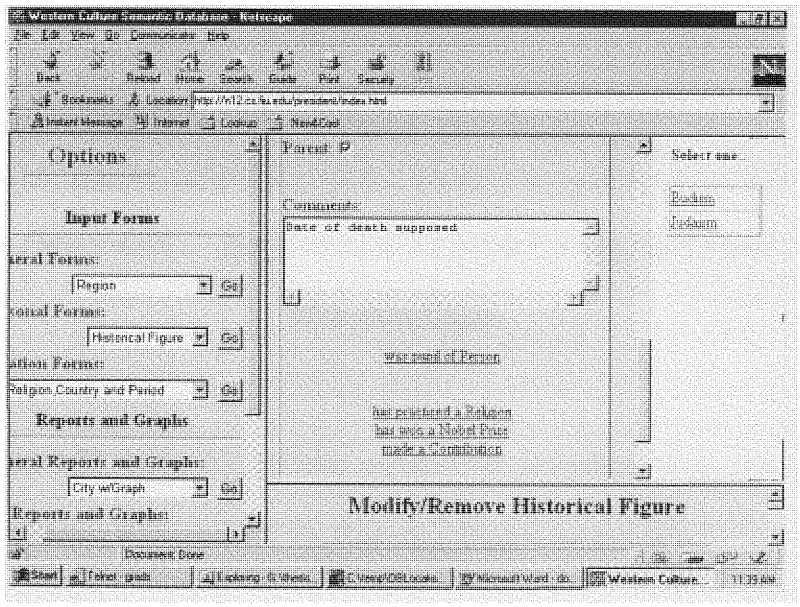


Figure 4.22 … continuation of Example 4.4 for: Input a new Historical Figure

After clicking the has practiced a Religion option, **Figure 4.22** shows the List of all the Religions in the database, so the user can pick one of them. Then because a Historical Figure can practice more than one Religion in his life, the user can click again the has practiced a Religion option and repeat the selection for another Religion.

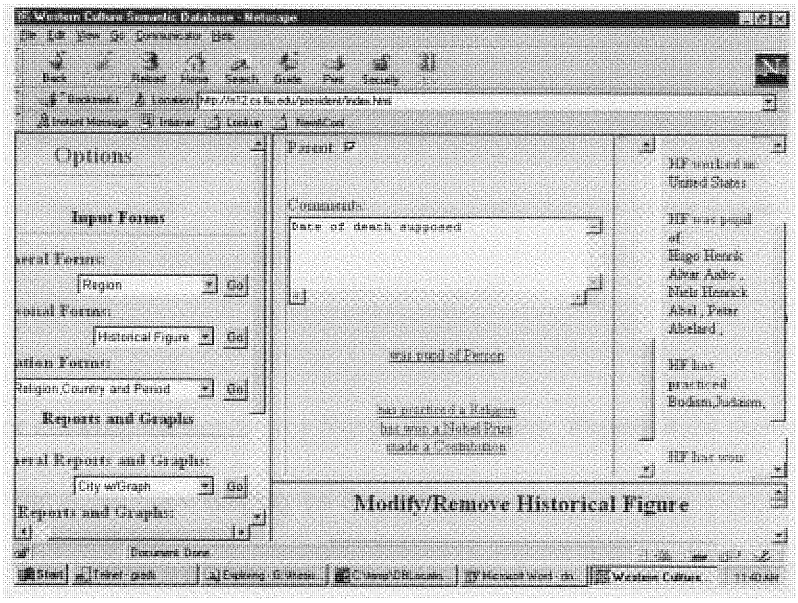


Figure 4.23 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.23 shows that the user repeated the procedure twice, and selected two Religions for the Historical Figure, to practice.

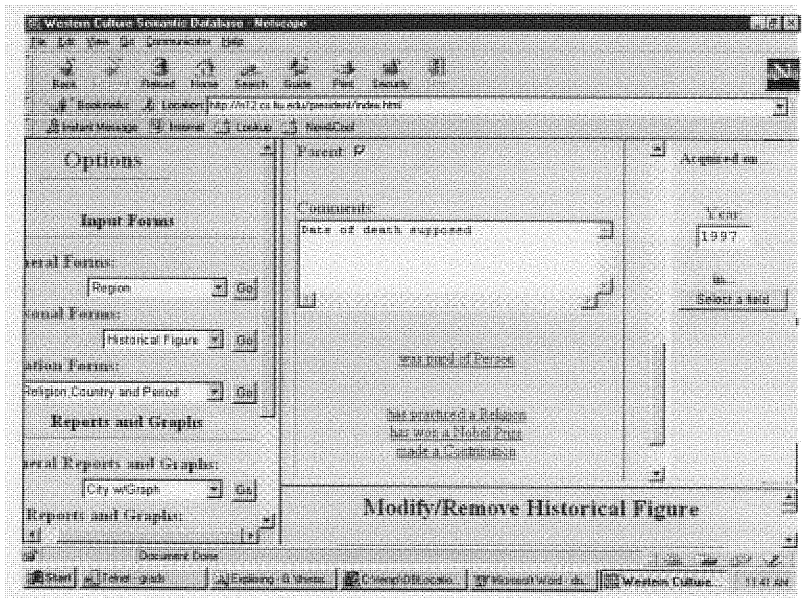


Figure 4.24 … continuation of Example 4.4 for: Input a new Historical Figure

After clicking the has won a Nobel Prize option, **Figure 4.24** shows that the application display on the right frame, a section to insert that the Nobel Prize was Acquired on… a particular year, and a button for the user to click and then select a Nobel Prize Field.

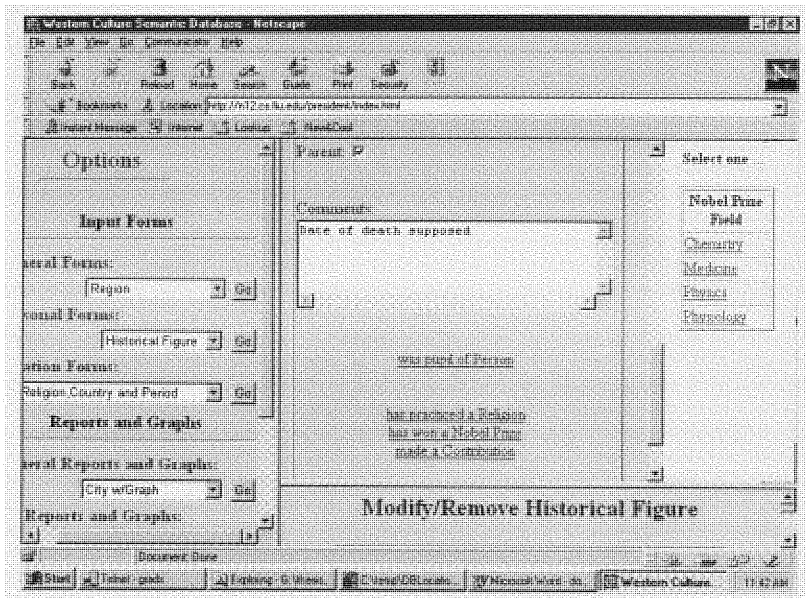


Figure 4.25 … continuation of Example 4.4 for: Input a new Historical Figure

After clicking the button, **Figure 4.25** shows the display of the List of all the Nobel Prize Fields in the database, so the user can pick one of them. Then because a Historical Figure can win more that one Nobel Prize in his life, the user can click again the has won a Nobel Prize option and repeat the insertion of another year and selection for another Prize for that year.

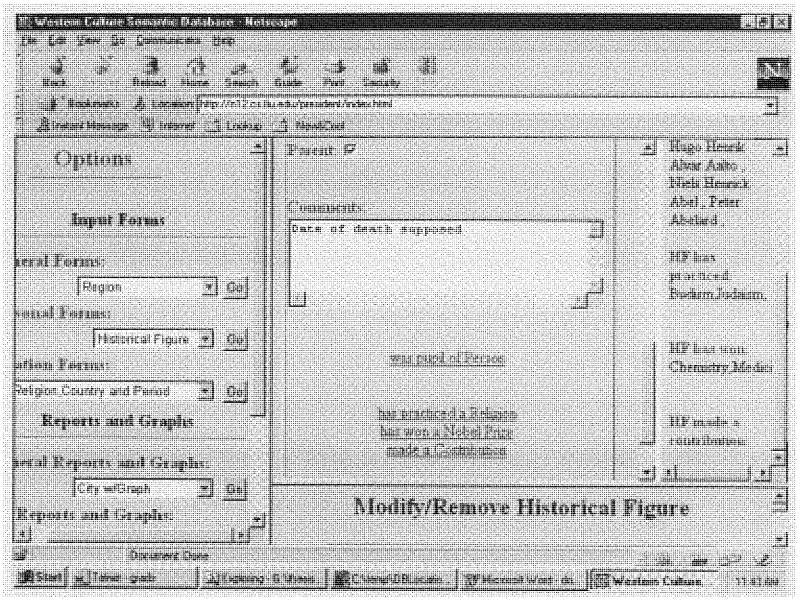


Figure 4.26 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.26 shows the relation of two prizes for the Historical Figure.

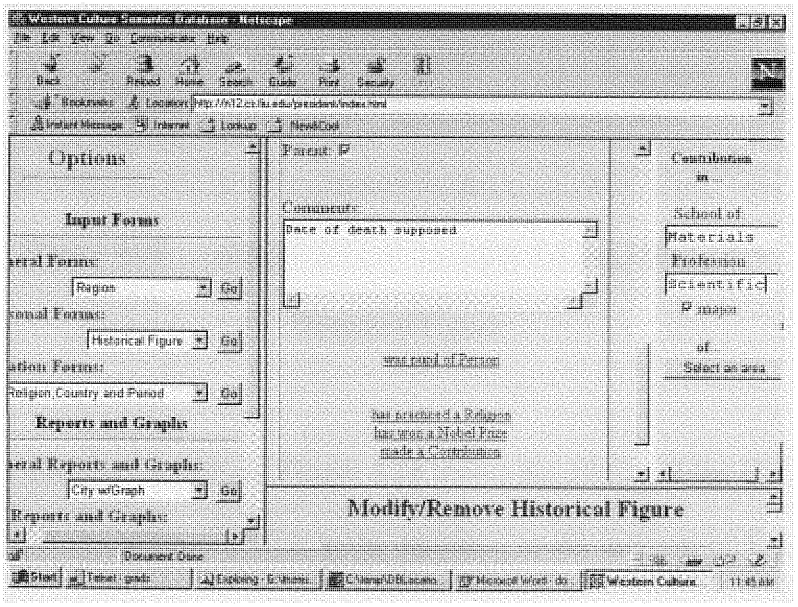


Figure 4.27 … continuation of Example 4.4 for: Input a new Historical Figure

After clicking the made a Contribution option, **Figure 4.27** shows that the application display on the right frame, a section to insert that the Contribution was in … a particular school, and in a particular profession. Also a button for the user to click and then select an Area of Contribution.

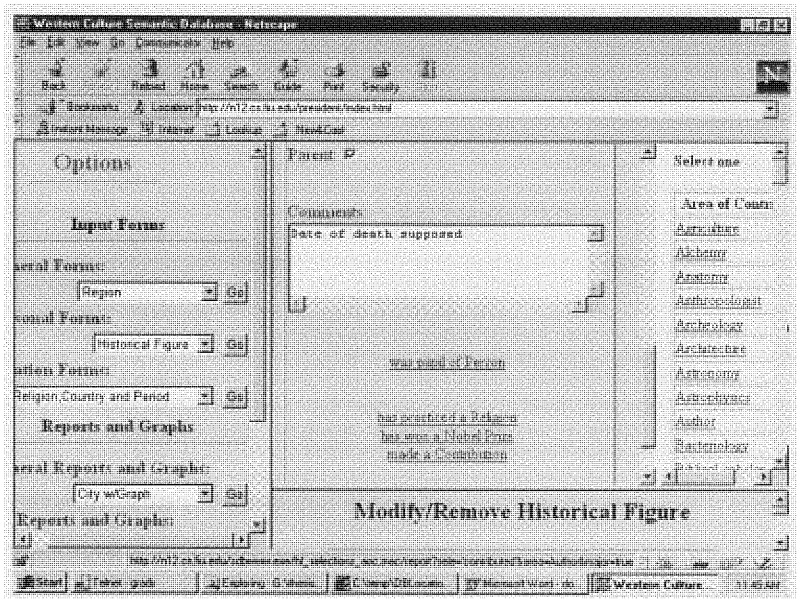


Figure 4.28 … continuation of Example 4.4 for: Input a new Historical Figure

After clicking the button, **Figure 4.28** shows the display of the List of all the Areas of Contribution in the database, so the user can pick one of them. Then because a Historical Figure make a contribution to more than one area , the user can click again the made a Contribution option

and repeat the insertion of another school, another profession, and selection for another Area for that contribution.

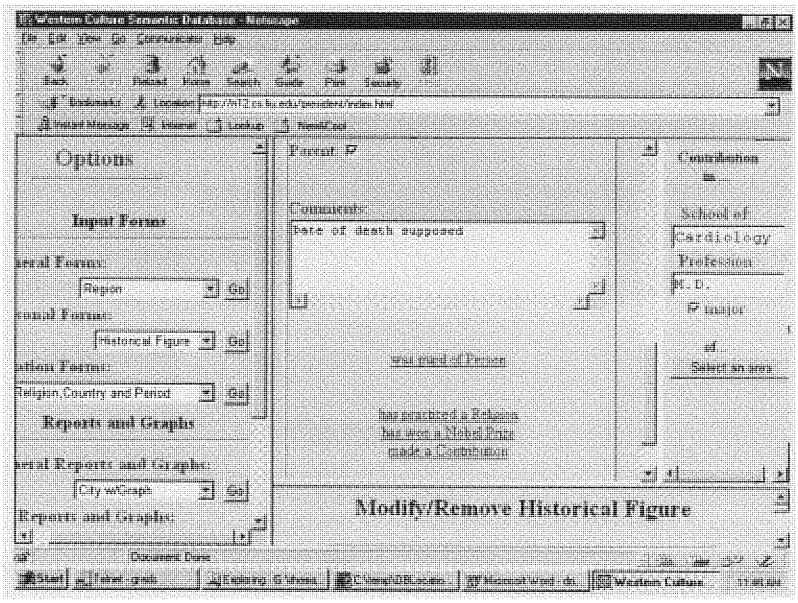


Figure 4.29 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.29 shows that the user made another Contribution.

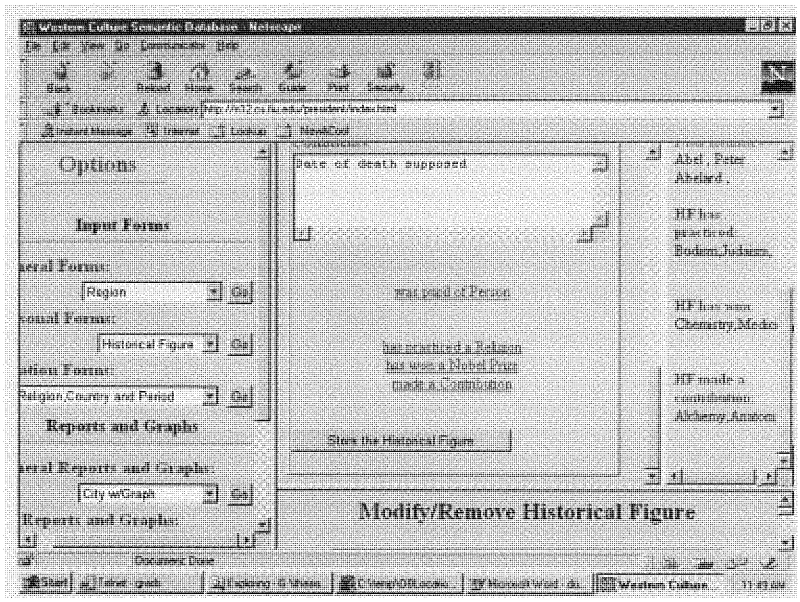


Figure 4.30 … continuation of Example 4.4 for: Input a new Historical Figure

Figure 4.30 shows that the user repeated the procedure twice, and related two Contributions for the Historical Figure, to make. See Selected values…(right frame).

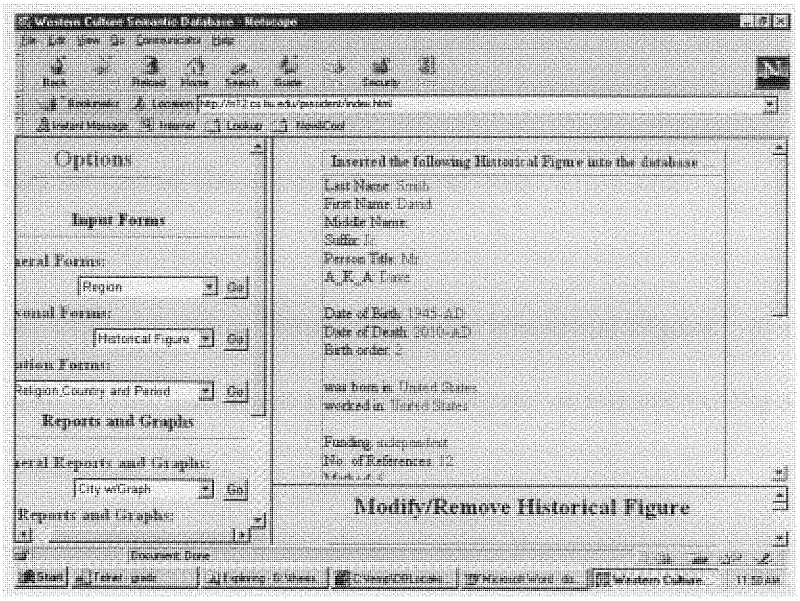


Figure 4.31 … continuation of Example 4.4 for: Input a new Historical Figure, Confirmation screen.

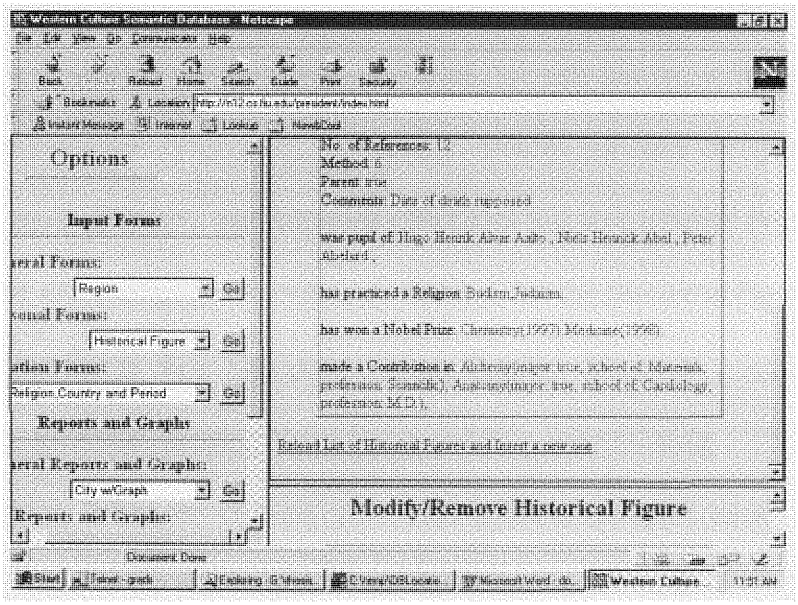


Figure 4.32 … continuation of Example 4.4 for: Input a new Historical Figure, Confirmation screen

Figure 4.31 and Figure 4.32 show a confirmation screen that displays the new data inserted into the database. Also gives the option of Reload the List of Historical Figures and Insert a new one.

The lower section is called "**Modify/Remove Historical Figure**", where the user can modify data about a Historical Figure, or remove the Figure from the database.

To Modify the place of birth and place of work, the user just has to click one of the options: was born in City or was born in Country, worked in City or worked in Country. The application displays the previous data for the user's reference. Moreover, For the Modify option the user can eliminate an existing m:m relation of the Historical Figure or add new one. To add new relations for the Historical Figure, the user just has to follow the same procedure as he did in the insertion section; just click the options for was pupil of Person, has practiced a Religion, has won a Nobel Prize, and made a Contribution. To eliminate relations for the Historical Figure, the user just has to click: Eliminate existing Teacher, Eliminate existing Religion, Eliminate existing Nobel Prize, and Eliminate existing Contribution. When any of those options are selected, the application displays a list for the user to pick which fact (relation) he wants to eliminate about the Historical Figure

Also, all the selected objects to be related to the Historical Figure, appear in the right hand side frame called Selected values....

Finally the user just click on the button: Store the Historical Figure, and the application gives a confirmation for the data entered in the database. See the Example 4.5 from **Figure 4.33** to **Figure 4.50**.

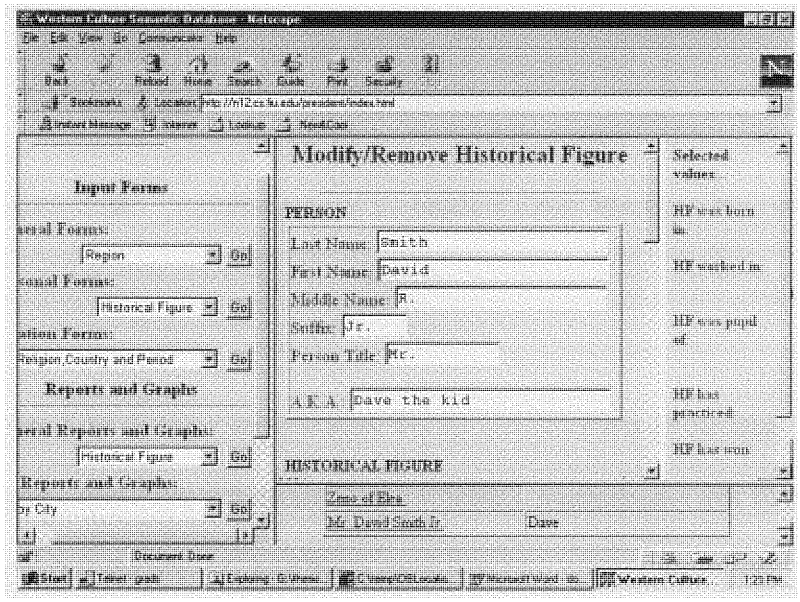


Figure 4.33 Example 4.5 for: Modify a Historical Figure,

Figure 4.33 shows the modification of the Historical Figure's data entry about his Middle Name and alias (A.K.A).

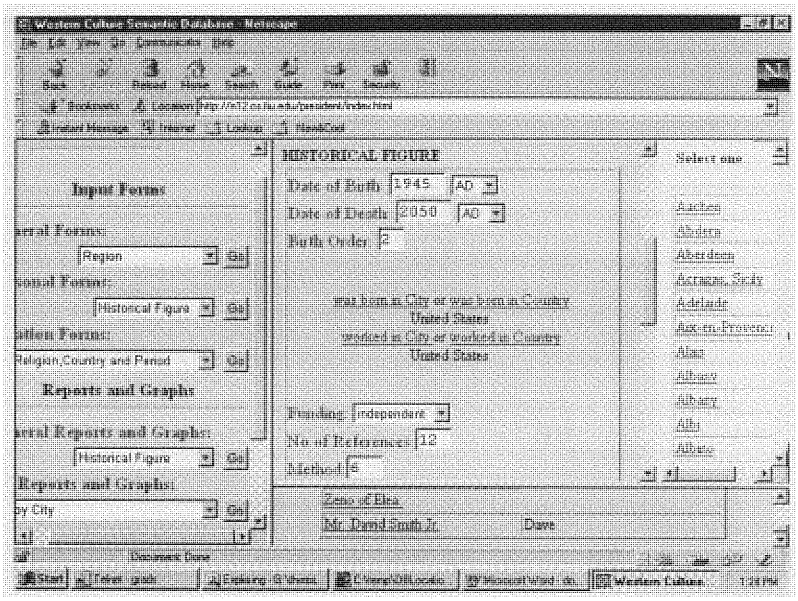


Figure 4.34 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.34 shows a continuation for the modification of the Historical Figure's data entry (about his Date of Death). Furthermore this Figure, shows that the user clicked the <u>was born in City</u> option, probably because he knows in which city the Figure was born. A list of all the cities in the database is presented on the right screen, to help him with the selection of the City of birth.

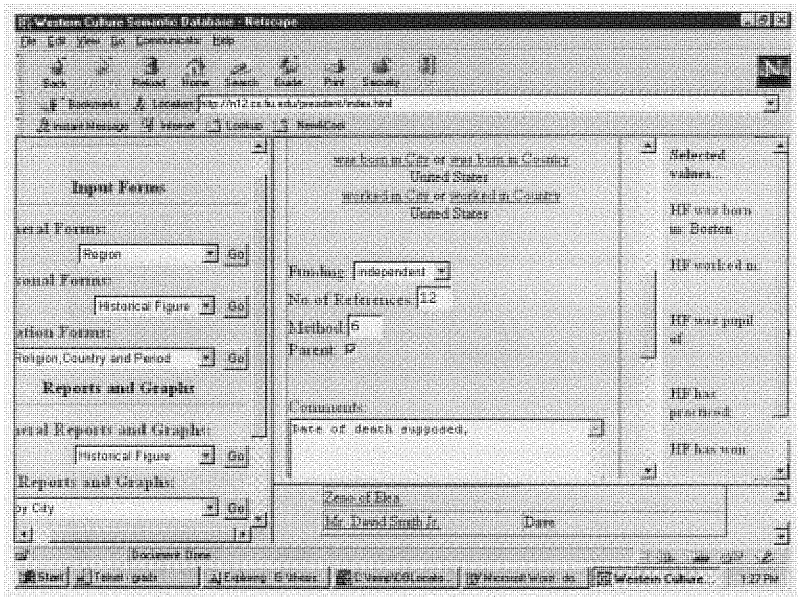


Figure 4.35 … continuation of Example 4.5 for: Modify a Historical Figure

After selecting the City where the Historical Figure was born, **Figure 4.35** shows the Selected values … on the right frame, at that particular time.

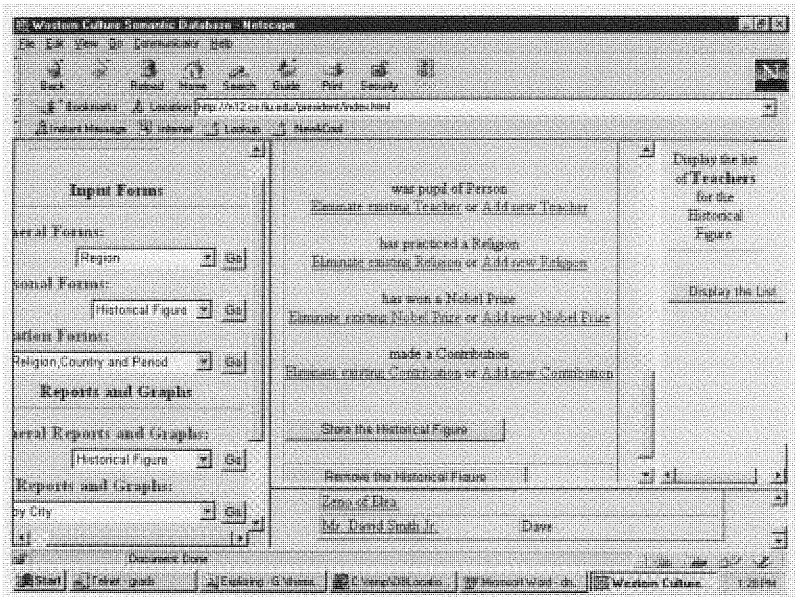


Figure 4.36 … continuation of Example 4.5 for: Modify a Historical Figure

After clicking the Eliminate existing Teacher option, **Figure 4.36** shows a button for the user to click, to display then List of all Persons that are teachers of the Historical Figure.

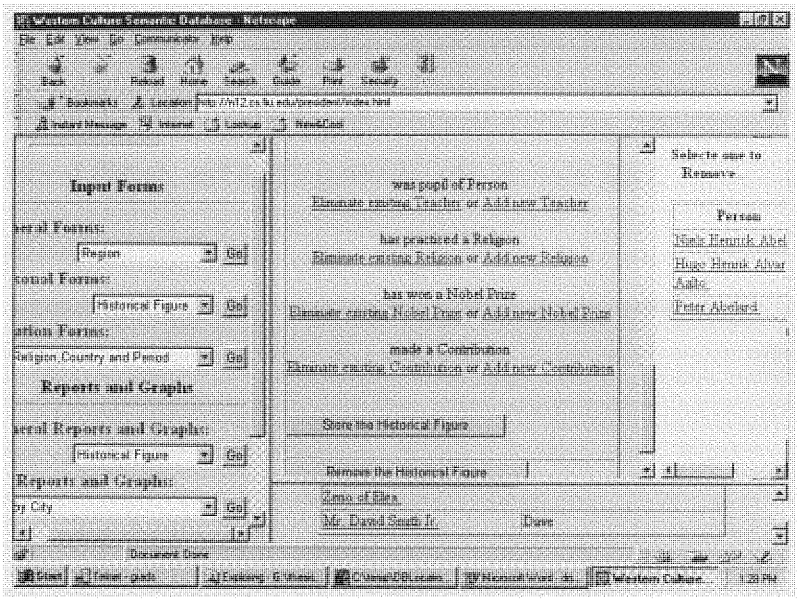


Figure 4.37 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.37 shows then a List of all the teachers of the Figure.

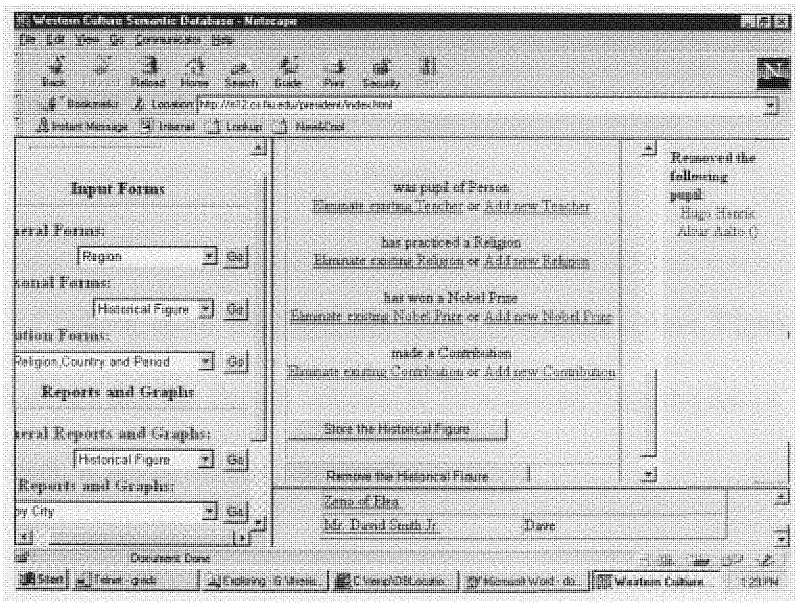


Figure 4.38 … continuation Example 4.5 for: Modify a Historical Figure

Figure 4.38 shows the selected person, that is not a teacher of the Historical Figure, and that's probably why the user wants to eliminate his relation to the Figure.

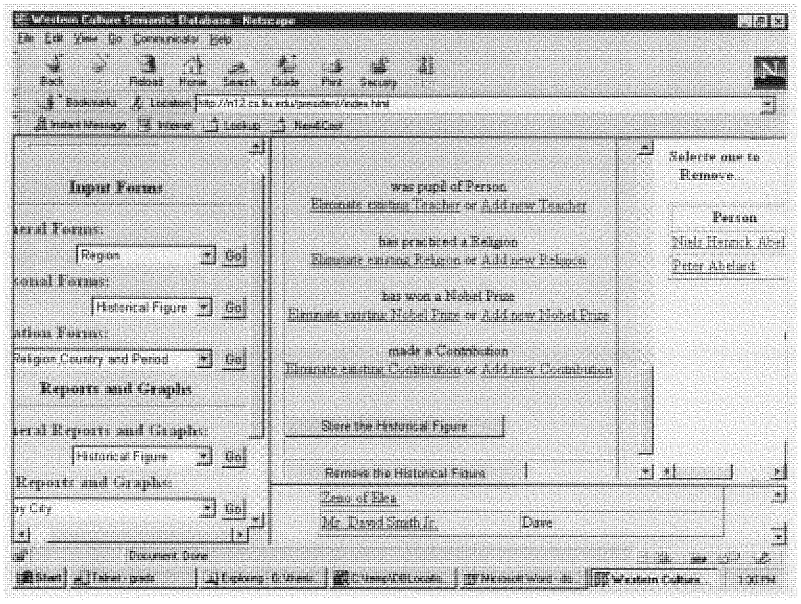


Figure 4.39 … continuation of Example 4.5 for: Modify a Historical Figure

After Clicking again, the Eliminate existing Teacher option, **Figure 4.39** shows that the Person selected to be removed from the list of teachers, is not in the list displayed.

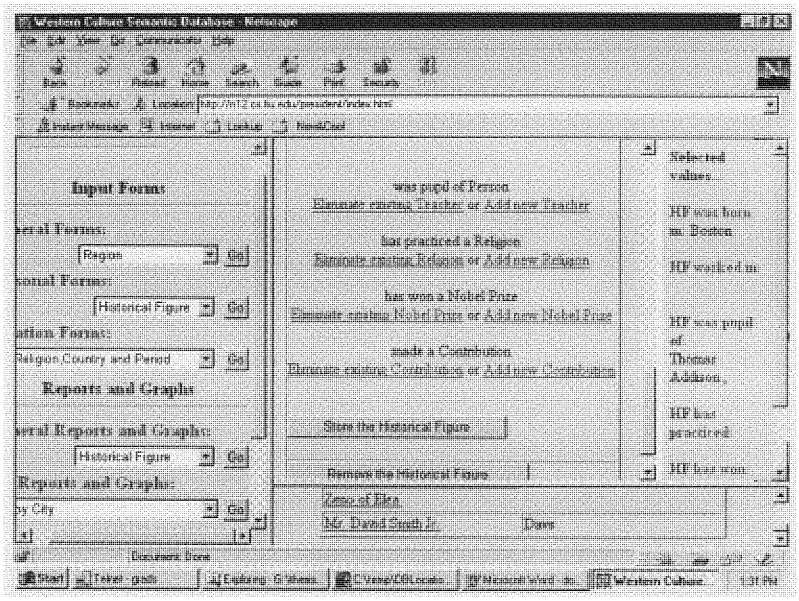


Figure 4.40 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.40 shows that the user clicked the option Add new Teacher, selected the teacher from the list of Persons, and the selection is presented in the right frame.

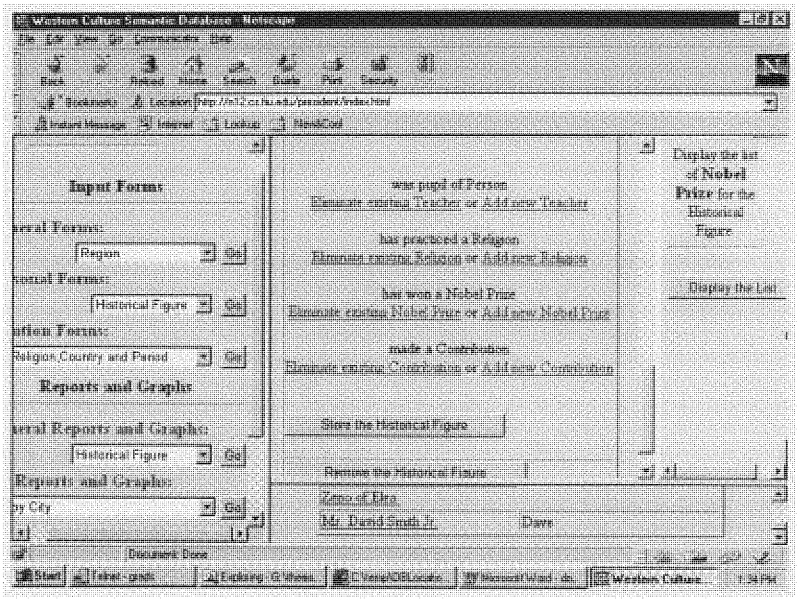


Figure 4.41 … continuation of Example 4.5 for: Modify a Historical Figure

After Clicking the Eliminate existing Nobel Prize option, **Figure 4.41** shows a button for the user to click, to display the List of all Nobel Prizes acquired by the Historical Figure.

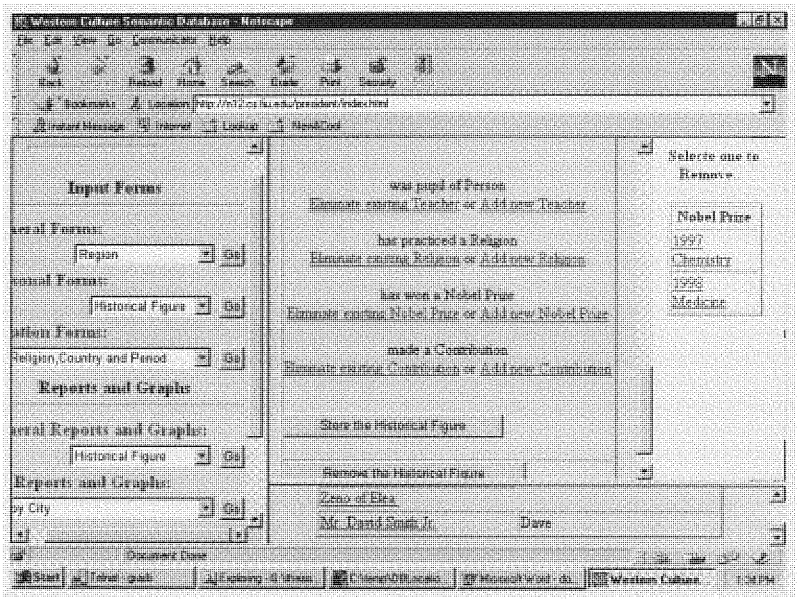


Figure 4.42 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.42 shows then a List of all the Prizes acquired by the Historical Figure. Then the user is able to to eliminate any from the list (probably because of a mistake, and the Historical Figure had not won a prize in that year).

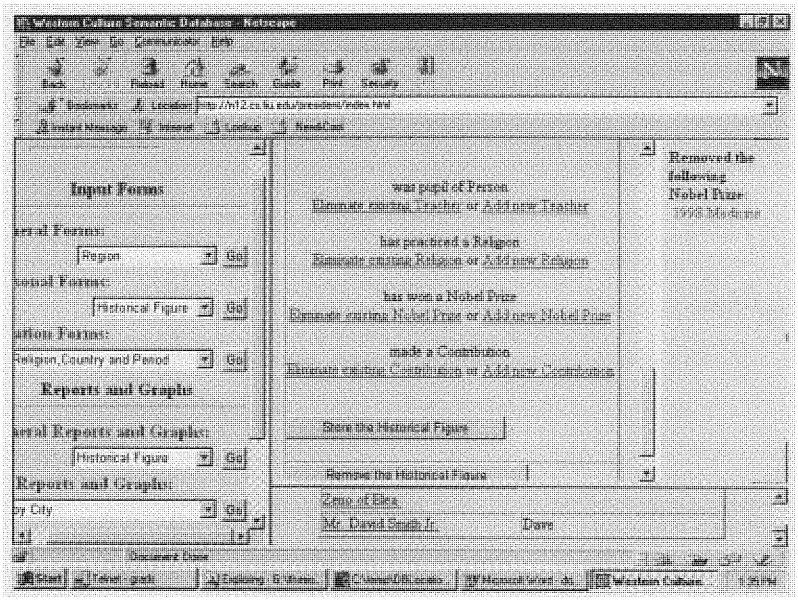


Figure 4.43 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.43 shows that the user selected, one Prize to eliminate, from the list.

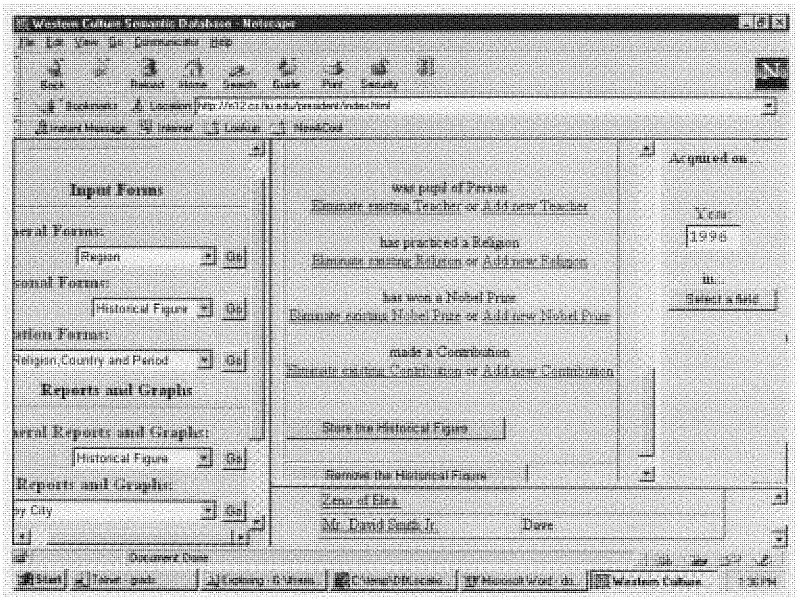


Figure 4.44 … continuation of Example 4.5 for: Modify a Historical Figure

Moreover, **Figure 4.44** shows that the user wants to add a new Nobel Prize acquired by the Historical Figure.

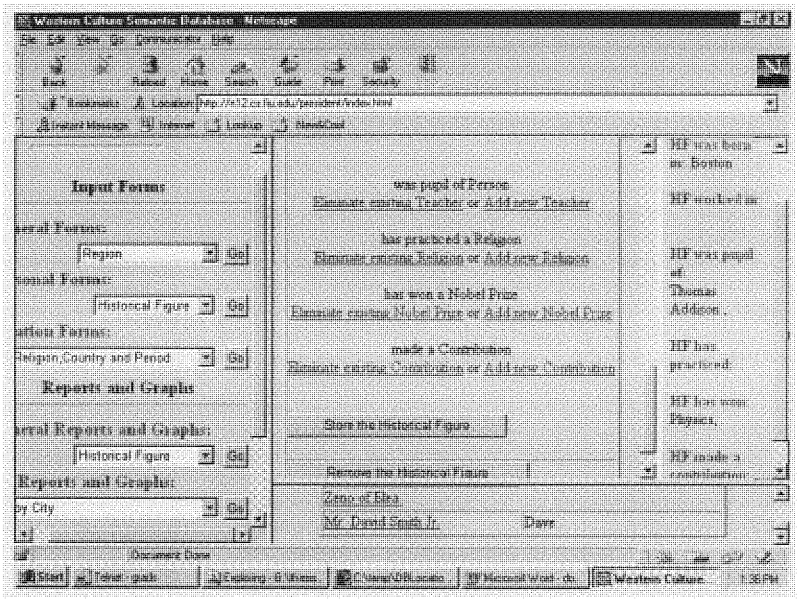


Figure 4.45 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.45 shows on the right hand frame, the selected values at that time.

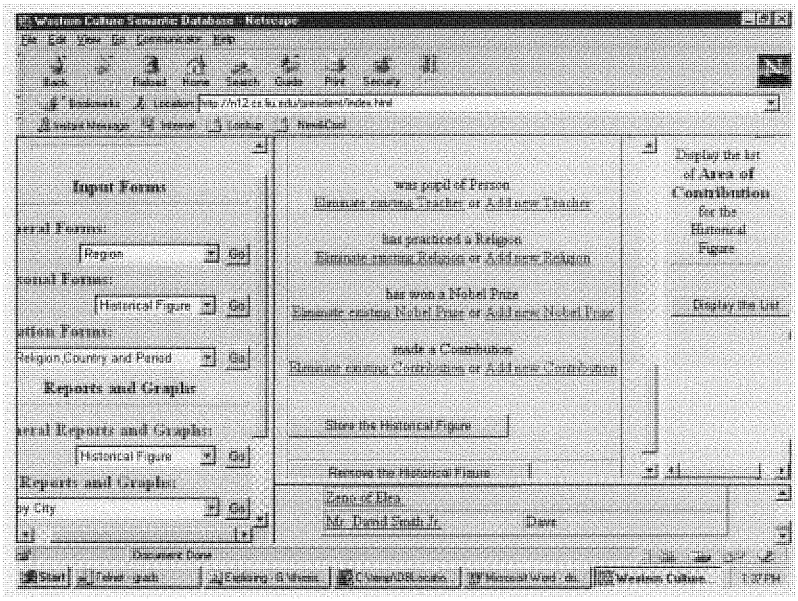


Figure 4.46 … continuation of Example 4.5 for: Modify a Historical Figure

In **Figure 4.46** the user wanted to Eliminate existing Contribution and the application gives the button for displaying the list of Contributions of that particular Historical Figure.

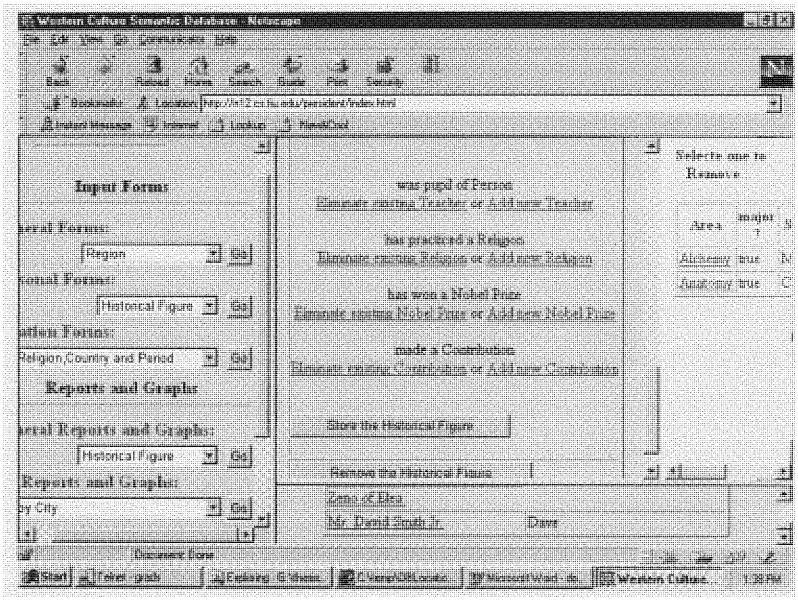


Figure 4.47 … continuation of Example 4.5 for: Modify a Historical Figure

In **Figure 4.47** is displayed the list of Contributions for that particular Historical Figure.

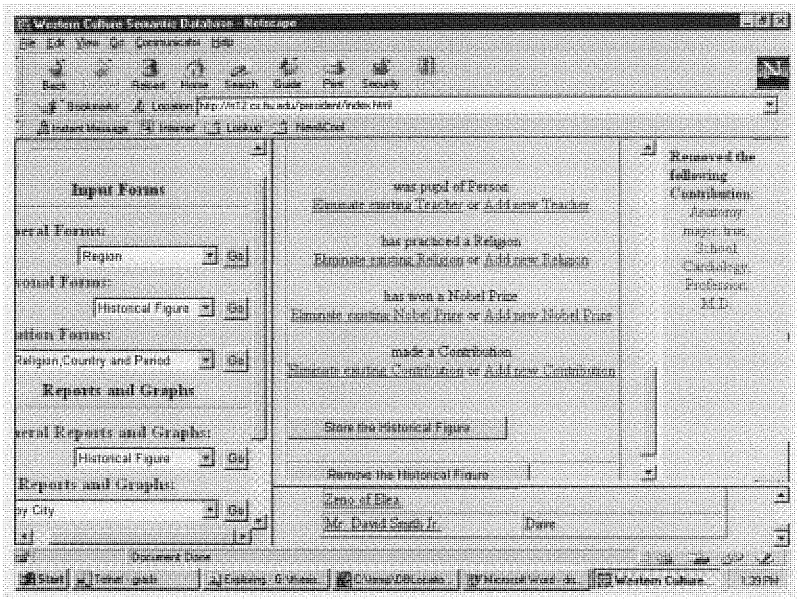


Figure 4.48 … continuation of Example 4.5 for: Modify a Historical Figure

Figure 4.48 shows the user's selection of the Contribution of the Historical Figure, to eliminate.

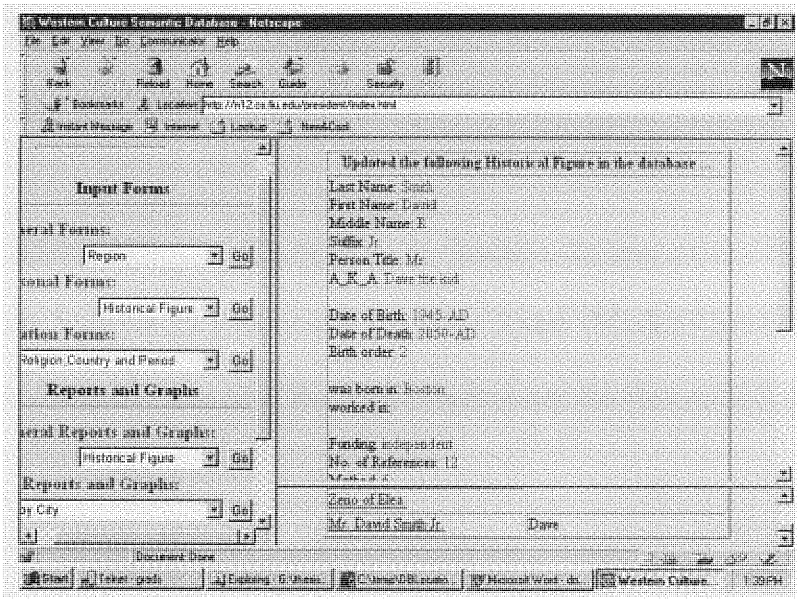


Figure 4.49 … continuation of Example 4.5 for: Modify a Historical Figure

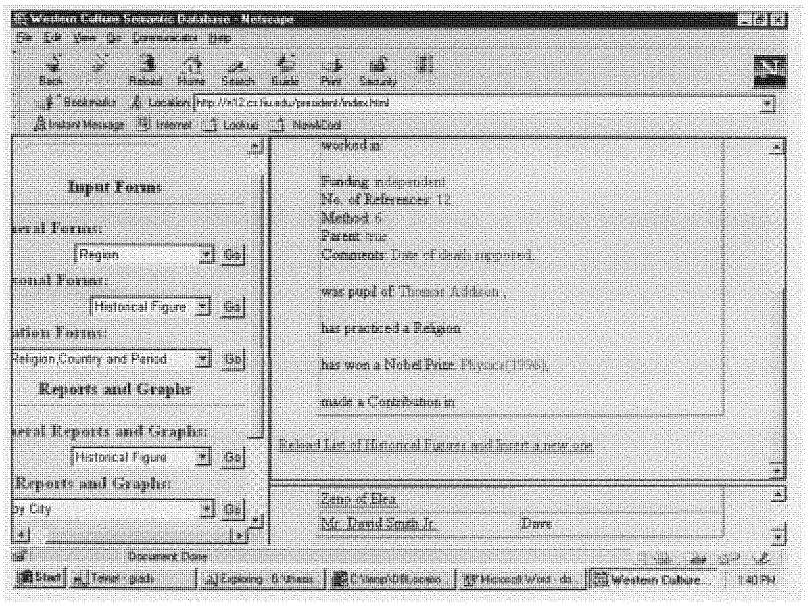


Figure 4.50 … continuation of Example 4.5 for: Modify a Historical Figure

Finally, **Figure 4.49** and **Figure 4.50** show the confirmation window for the modifications done by the user.

To Eliminate a Historical Figure from the database, the user must select the Figure from the list presented in the lower **frame (Modify/Remove Historical Figure**), and click the button for Remove the Historical Figure.

- Relation Forms: A group of forms to relate other objects in the database. In the group is included: Religion Country and Period, Contribution to Area and Period, and Country and Population.

  - Religion, Country and Period: This option relates a Religion and a Country that are in the database, with a period that is inserted. The option divides the screen in two sections (like before), **the Input Religion-Country-Period relation** frame, and **the Modify/Remove a Period** frame.

    For the Input section, the user has to enter the period data, and select a Country and a Religion from the lists that are displayed in the screen. See **Figure 4.51**

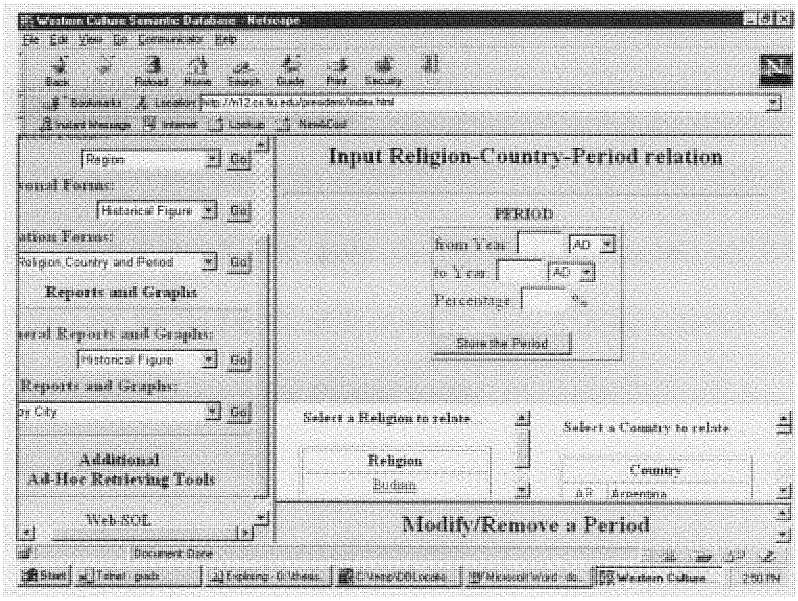


Figure 4.51 Religion, Country and Period Form – Relation Forms

For the Modify/Remove section, the user has to select a period from the list displayed in the lower frame, and then modify or remove it from the database. See **Figure 4.52**

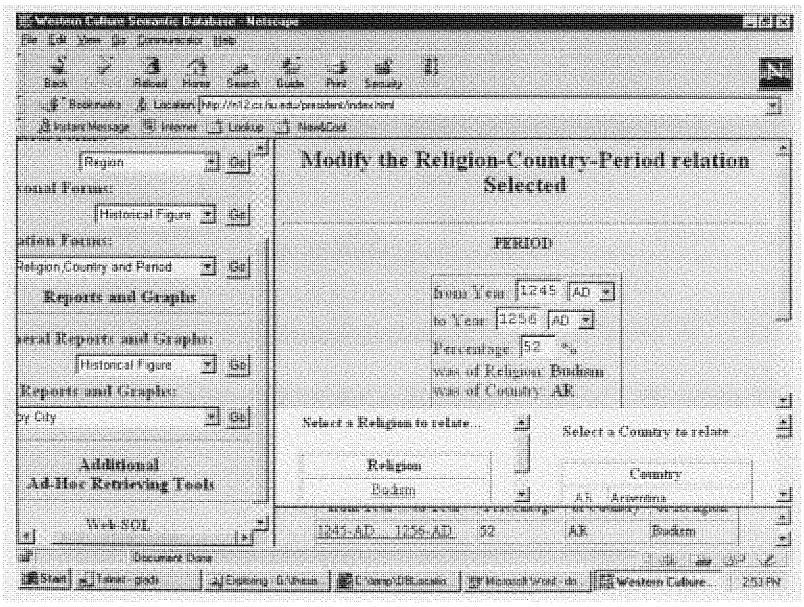


Figure 4.52 Modify/Remove Religion, Country and Period Form

- Contribution to Area and Period: This option relates a Contribution to Area that has being inserted (in the Historical Figure Form) with a period that is inserted. The option divides the screen in two sections (like before), **the Input Contribution to Area-Period relation** frame, and **the Modify/Remove a Period** frame.

  For the Input section, the user has to enter the period data, and select the Contribution from the list that is displayed in the screen. See **Figure 4.53.**

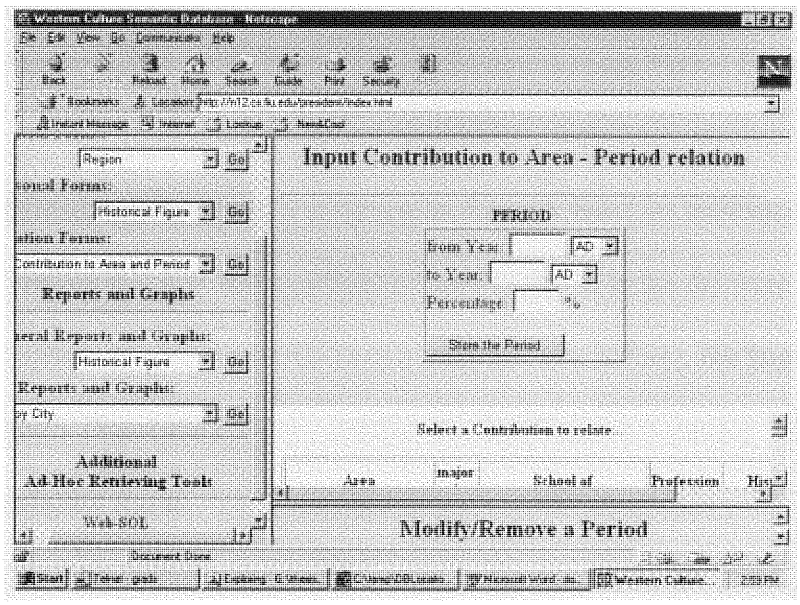


Figure 4.53 Contribution to Area and Period Form – Relation Forms

For the Modify/Remove section, the user has to select a period from the list displayed in the lower frame, and then modify or remove it from the database. See **Figure 4.54**

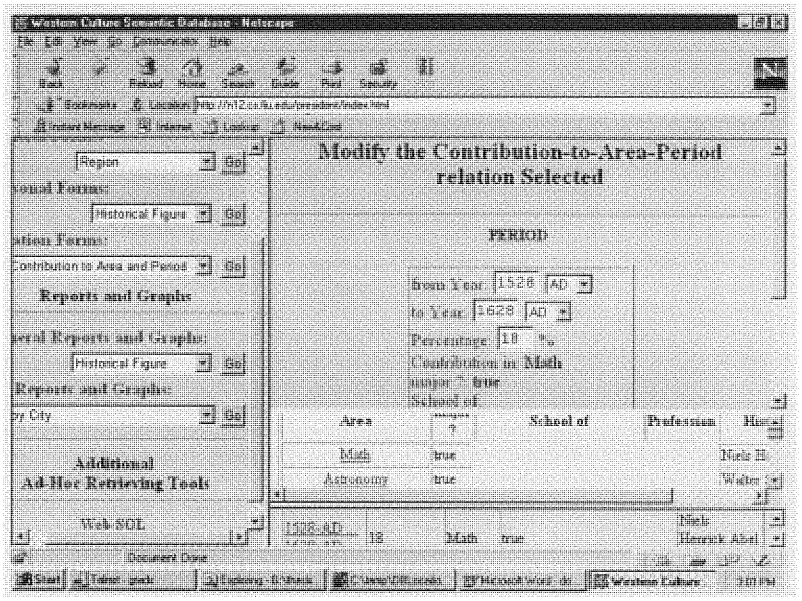


Figure 4.54 Modify/Remove Contribution-to-Area-Period Form

- Country and Population: This option let the user insert an estimated population in a year, for a particular Country that is in the database. The option divides the screen in two sections (like before), **the Input Country Population** frame, and **the Modify/Remove Country Population** frame.

  For the Input section, the user has to enter the year and population, and select the Country from the list. See **Figure 4.55.**

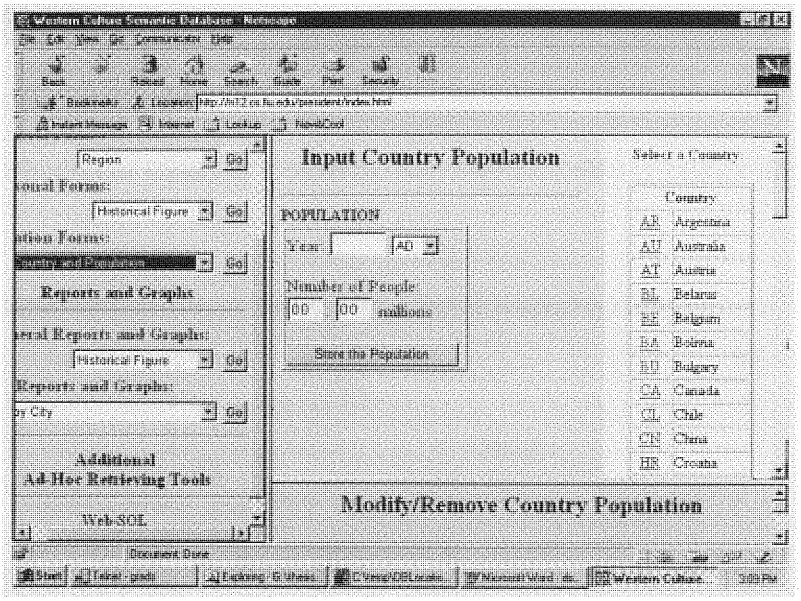


Figure 4.55 Country Population Form – Relation Forms

For the Modify/Remove section, the user has to select an item from the list displayed in the lower frame, and then modify or remove it from the database.

2. Reports and Graphs: Includes General reports to query some categories in the database, and some more complicated reports about Historical Figures, specially requested by the President's office. Also include some Graphs that generate valuable information about the data stored in the database.

   - General Reports and Graphs: A group of Reports and Graphs about general categories in the database.

- City w/Graph: This option generates a Report of all the cities in the database grouped by Country, and display a graph that reflects the distribution of Cities by Countries. See **Figure 4.56** and **Figure 4.57**

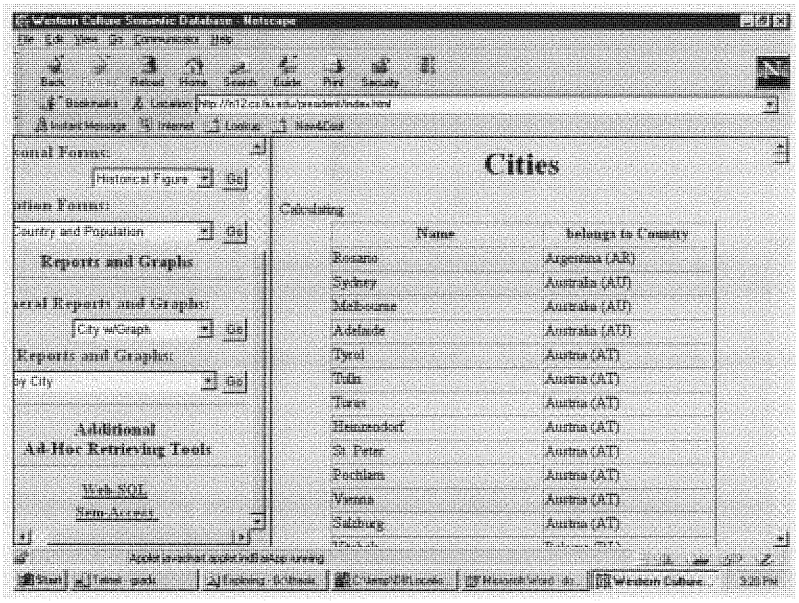


Figure 4.56 City w/Graph – General Reports and Graphs

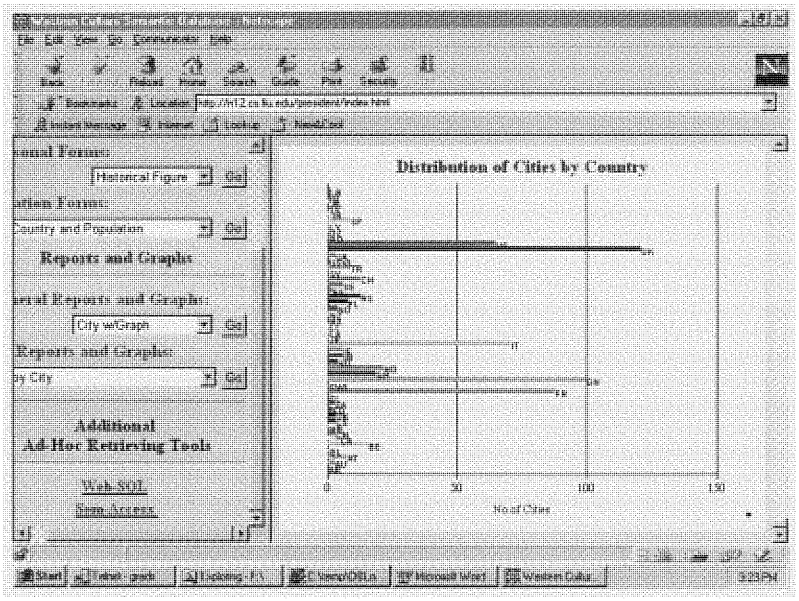


Figure 4.57 ...continuation City w/Graph – General Reports and Graphs

In **Figure 4.57** is reflected that the Country who has more cities in the database is United Kingdom (UK), followed by Germany (GN) and France (FR). This happens because probably the database contains more Historical Figures that were born in this Country[10].

- Person: This option generates a Report of all the persons in the database.
- Historical Figure: This option generates a Report of all the Historical Figures in the database. It includes the option of selecting a particular Figure and displays his information. See **Figure 4.58**, **Figure 4.59** and **Figure 4.60**.

[10] There is no data provided by the President's office, about the place of work for the Historical Figures.

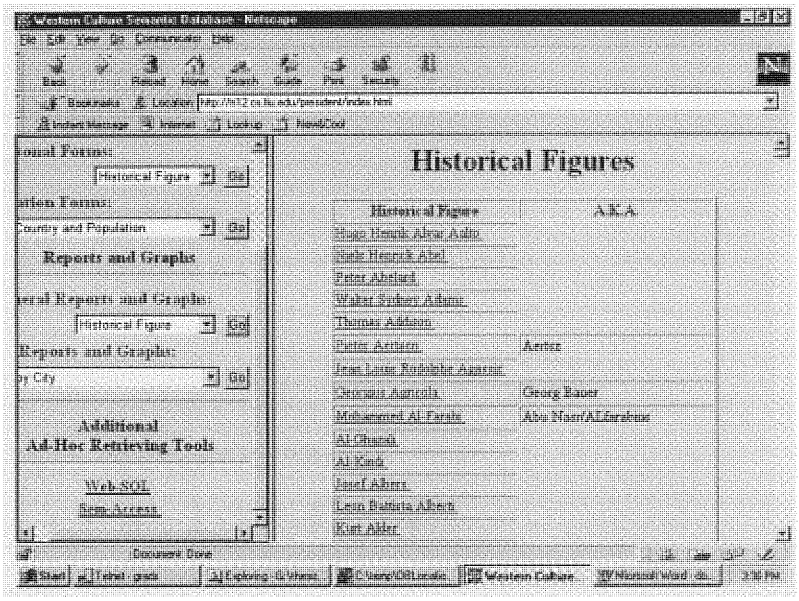


Figure 4.58 Historical Figure – General Reports and Graphs

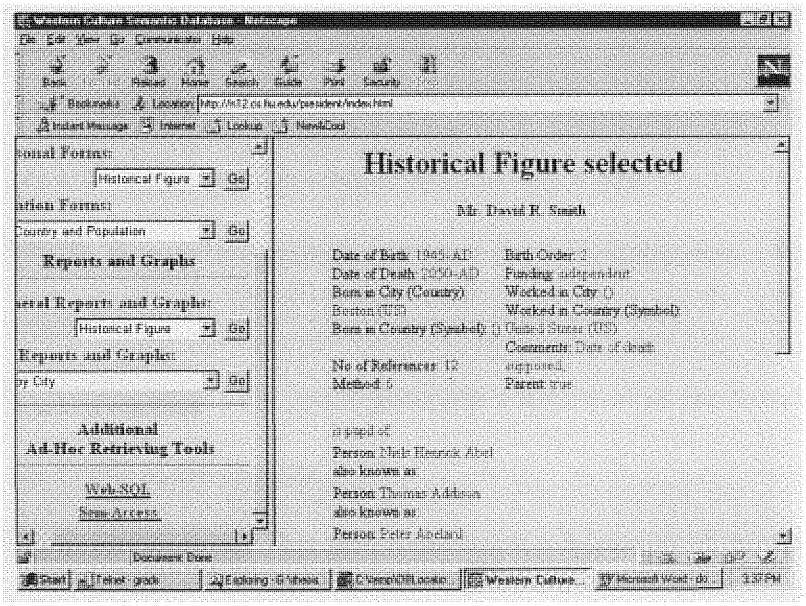


Figure 4.59 Historical Figure selected – General Reports and Graphs

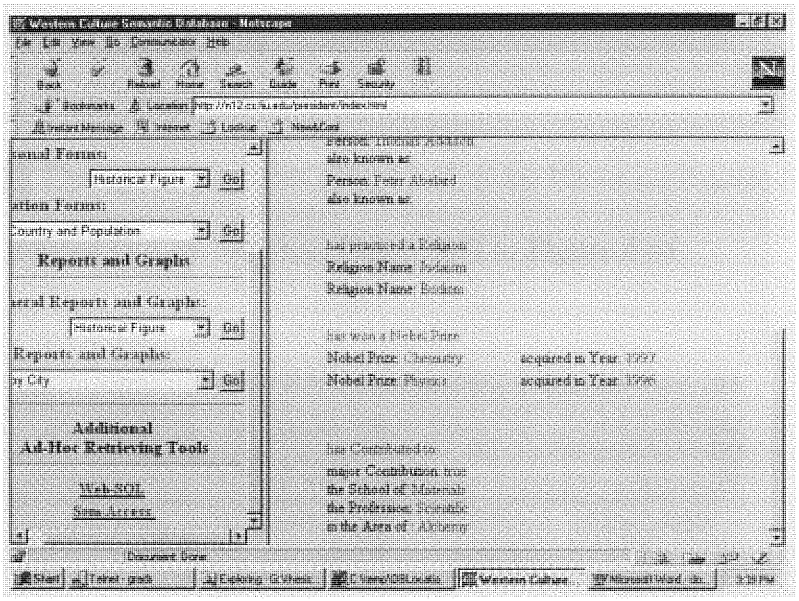


Figure 4.60 … continuation Historical Figure selected – General Reports and Graphs

Figure 4.59 and **Figure 4.60** show the Historical Figure inserted in Example 4.4 after his modification done in Example 4.5.

- Country: This option generates a Report of all the Countries in the database.
- Nobel Prize Field: This option generates a Report of all the Nobel Prize Fields in the database.
- Area of Contribution: This option generates a Report of all the Areas of Contribution in the database.

- HF Reports and Graphs: A group of Reports and Graphs about Historical Figures stored in the database. For this Graphs the application gives the user the option to

display the graph for a small group of items (Cities, Countries, Areas of Contribution, or Nobel Prize Fields) for for all the elements in the database.

- HF by City: This option generates a Report of all the Historical Figures in the database grouped by City. See from **Figure 4.61** to **Figure 4.64**.

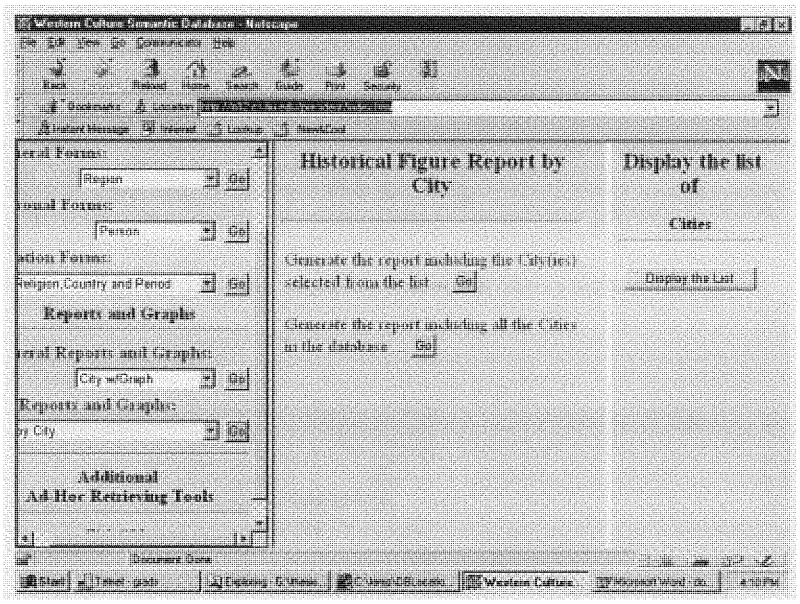


Figure 4.61 HF by City – HF Reports and Graphs

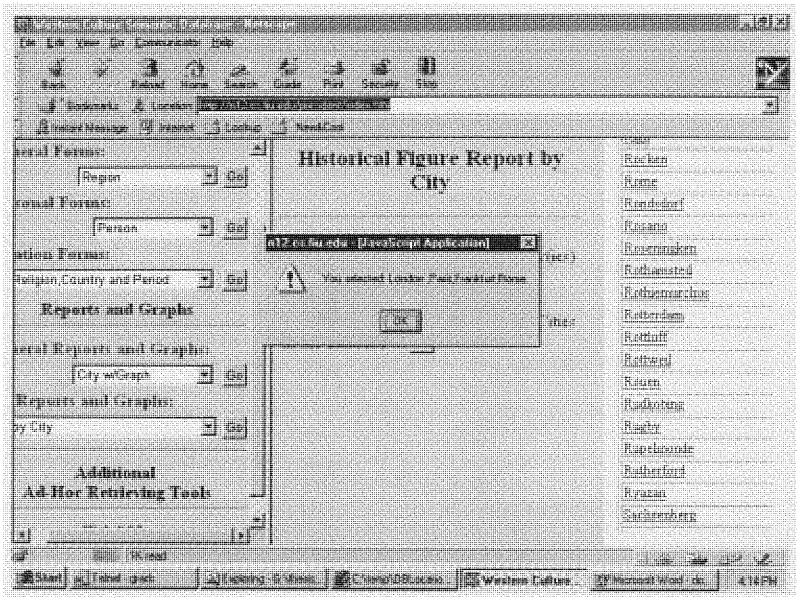


Figure 4.62 HF by City – Report for selected Cities

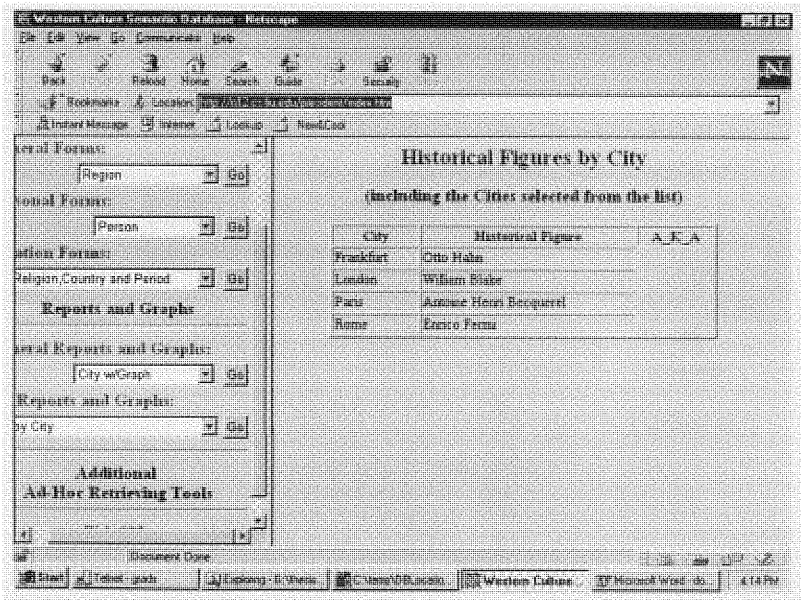


Figure 4.63 …continuation HF by City – Report for selected Cities

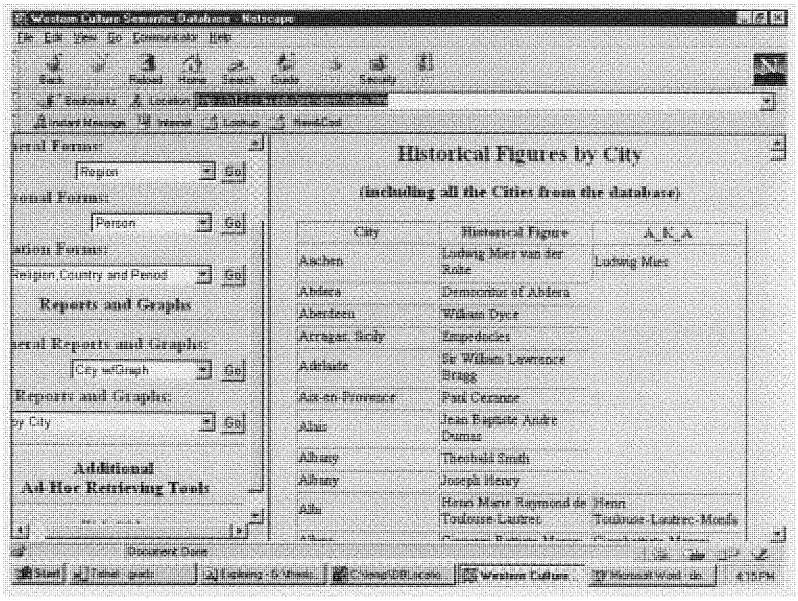


Figure 4.64 HF by City – Report for all the Cities in the database.

- HF by Country w/Graph: This option generates a Report of all the Historical Figures in the database grouped by Country. It also generates a graph of the data. See from **Figure 4.65** to **Figure 4.69**.

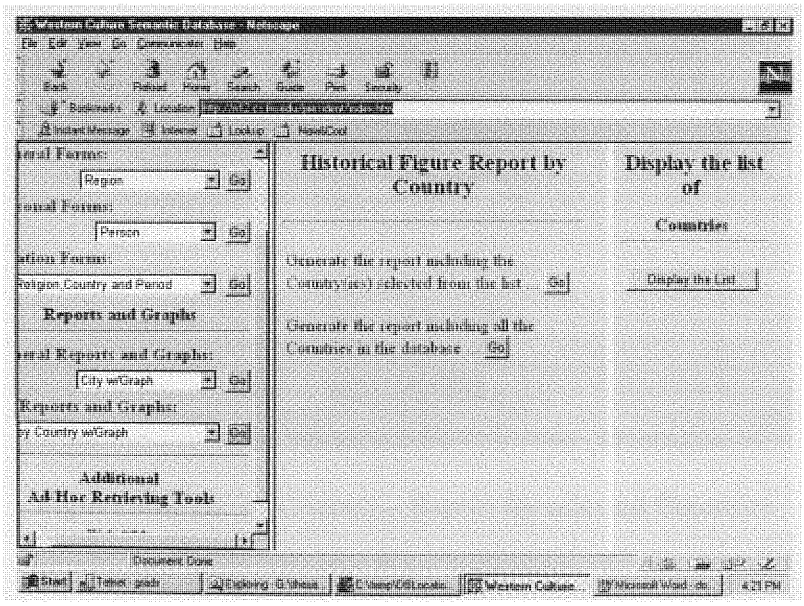
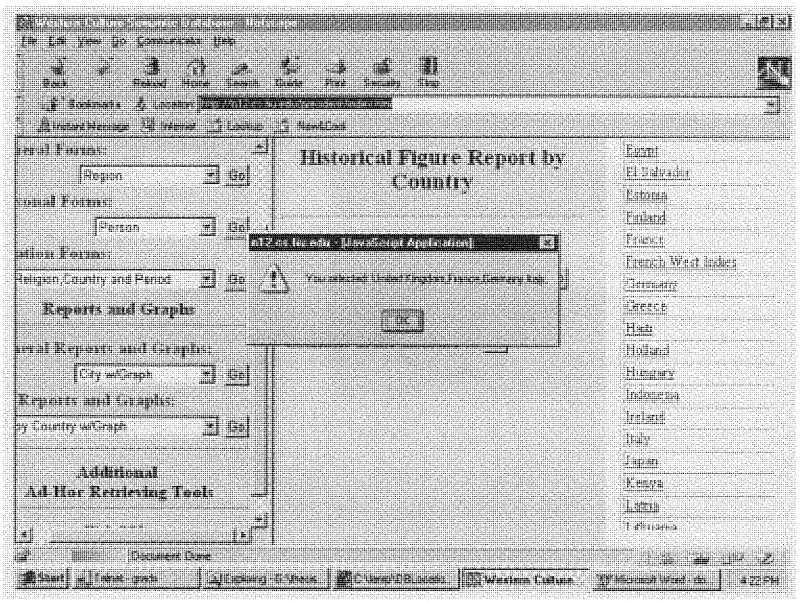


Figure 4.65 HF by Country – HF Reports and Graphs



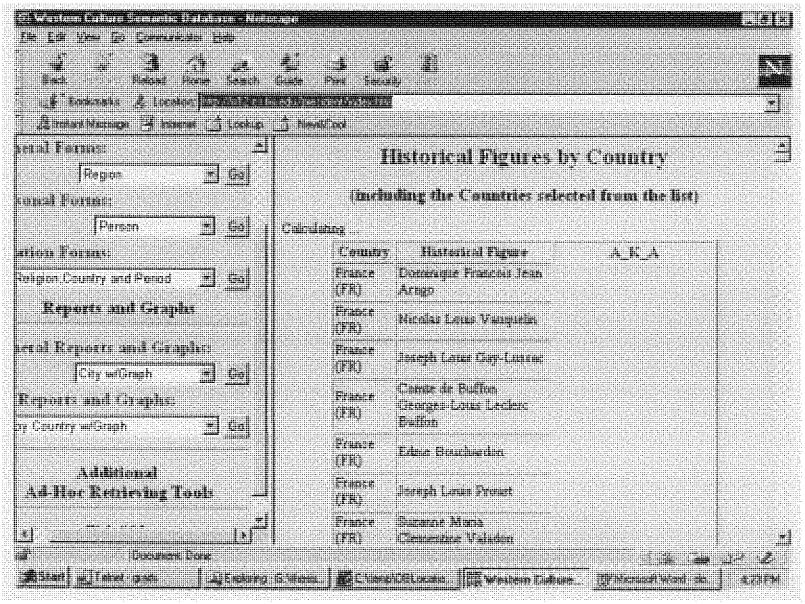


Figure 4.66 HF by Country – Report for selected Countries

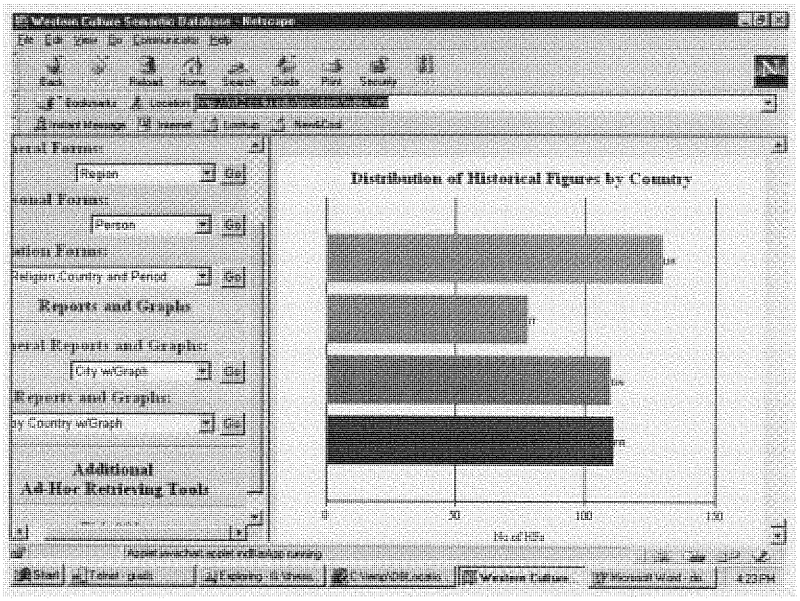


Figure 4.67 …continuation HF by City – Report for selected Cities

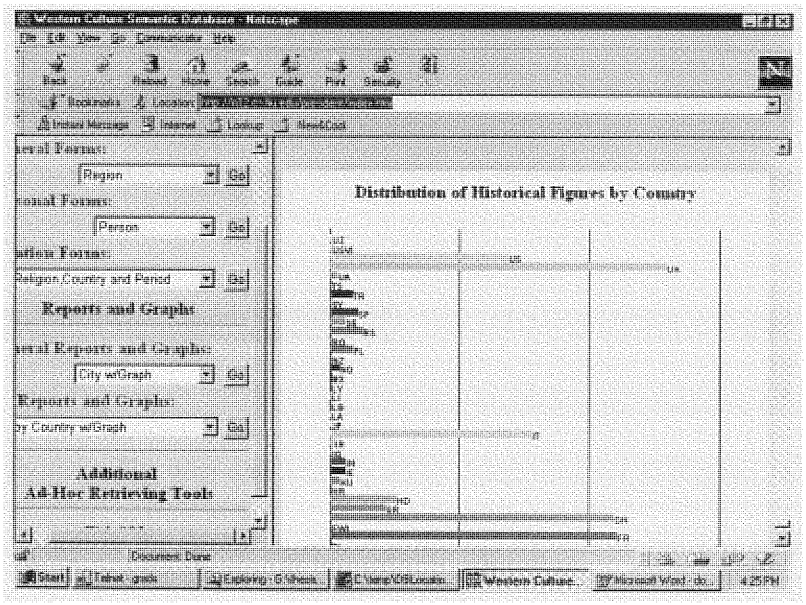


Figure 4.68 HF by Country – Report for all the Countries in the database.

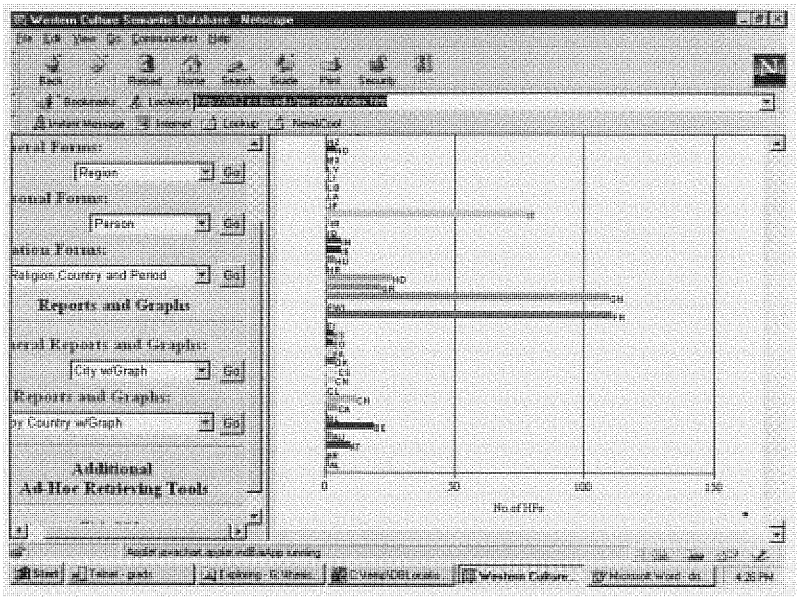


Figure 4.69 ...continuation HF by Country –

Report for all the Countries in the database.

- HF by Country and City: This option generates a Report of all the Historical Figures in the database grouped by Country, detailing the City where they were born.
- HF by Area of Contribution w/Graph: This option generates a Report of all the Historical Figures in the database grouped by major (or secondary) Area of Contribution. The user can select a small group of areas to display, or include all the Areas in the database. It also generates a graph of the data for each option. See from **Figure 4.70** to **Figure 4.74**.

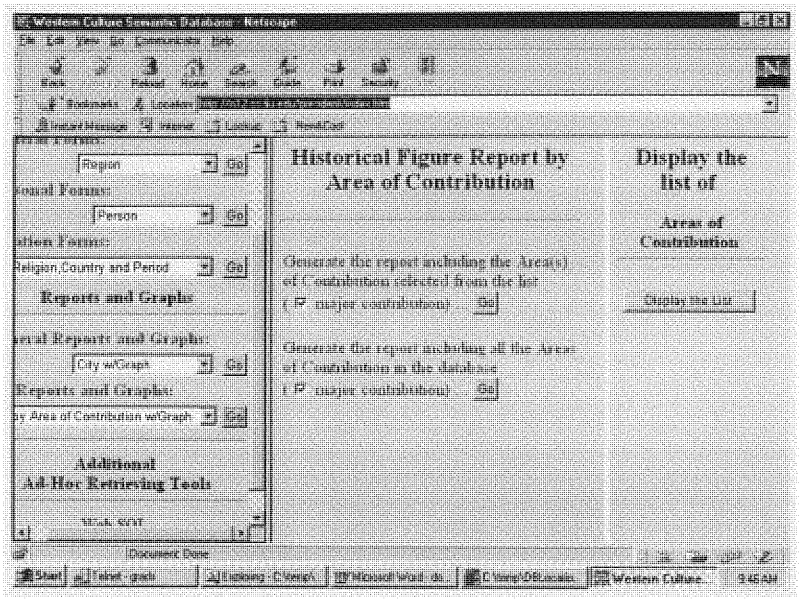


Figure 4.70 HF by Area of Contribution – HF Reports and Graphs

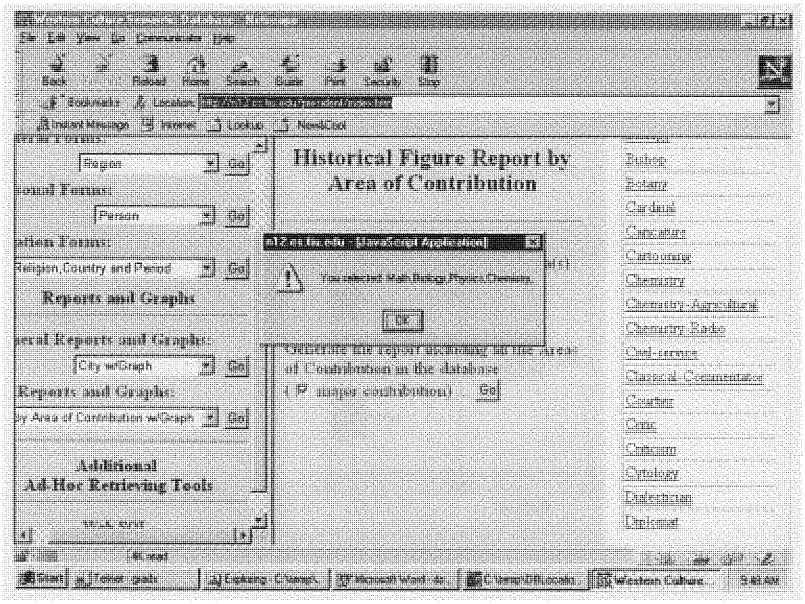
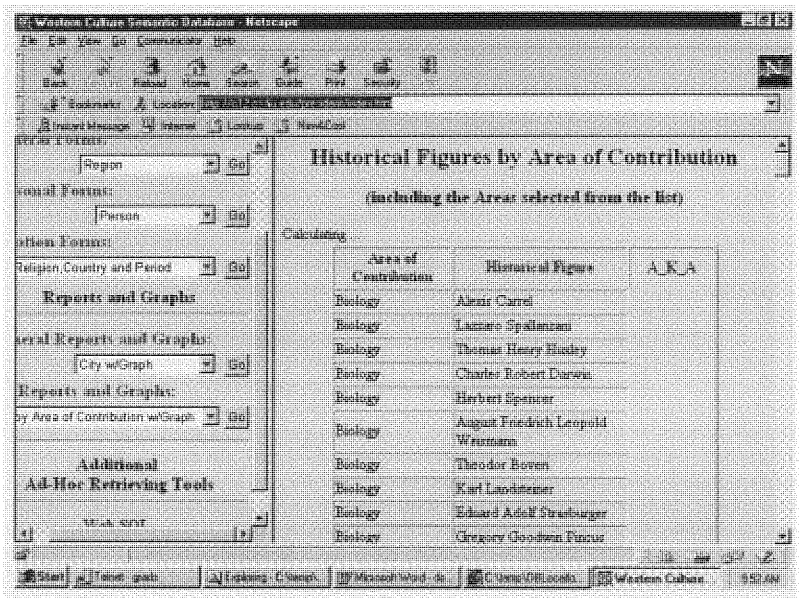


Figure 4.71 HF by Area of Contribution – Report for selected Areas



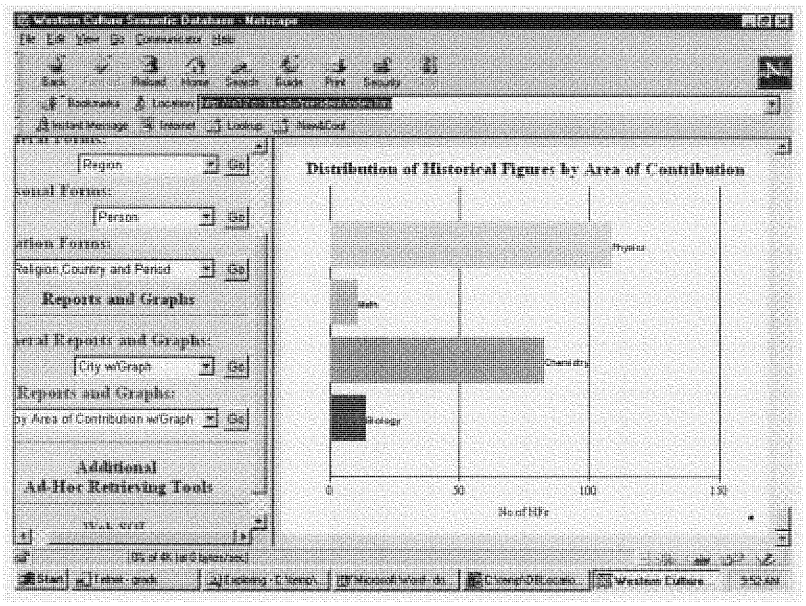


Figure 4.72 …continuation HF by Area of Contribution – Report and Graph for selected Areas

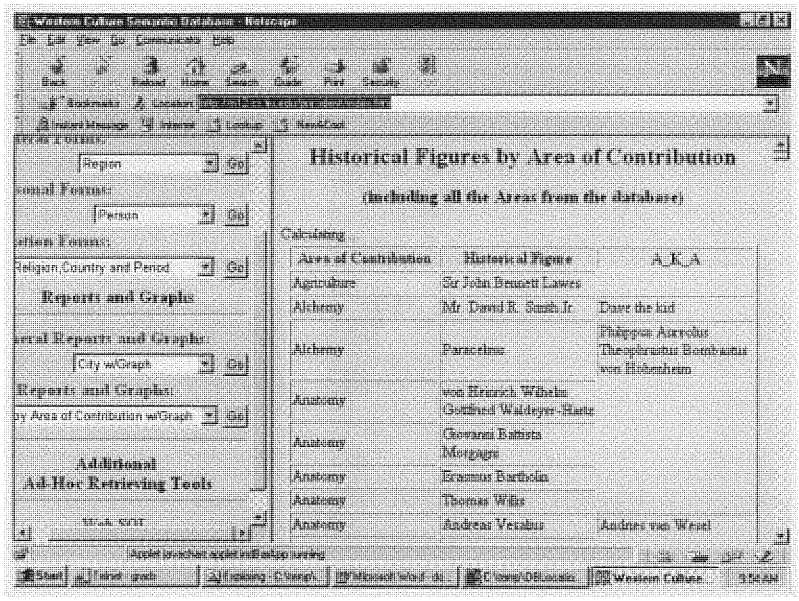
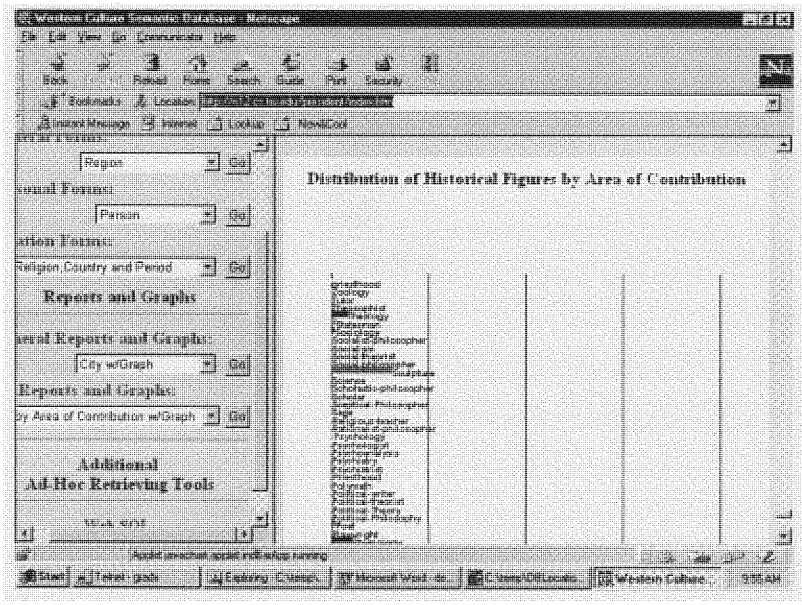
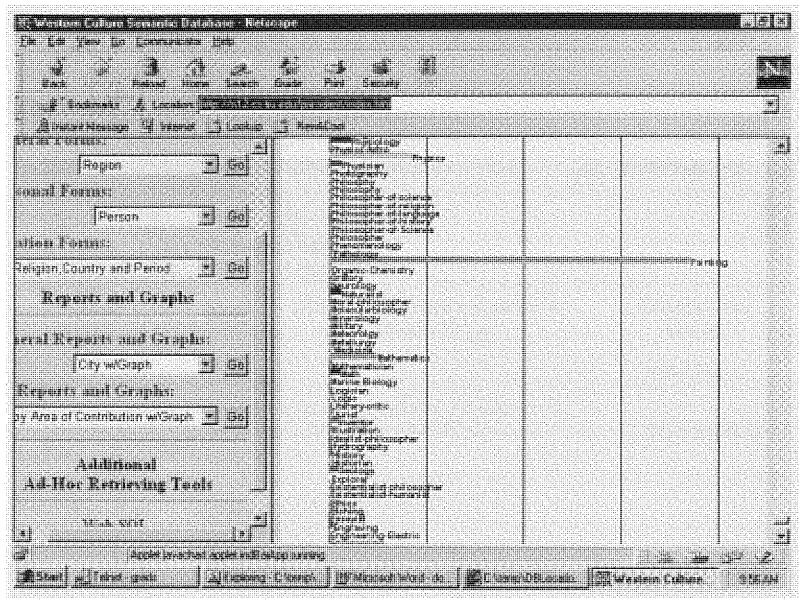


Figure 4.73 HF by Area of Contribution – Report for all the Areas in the database.





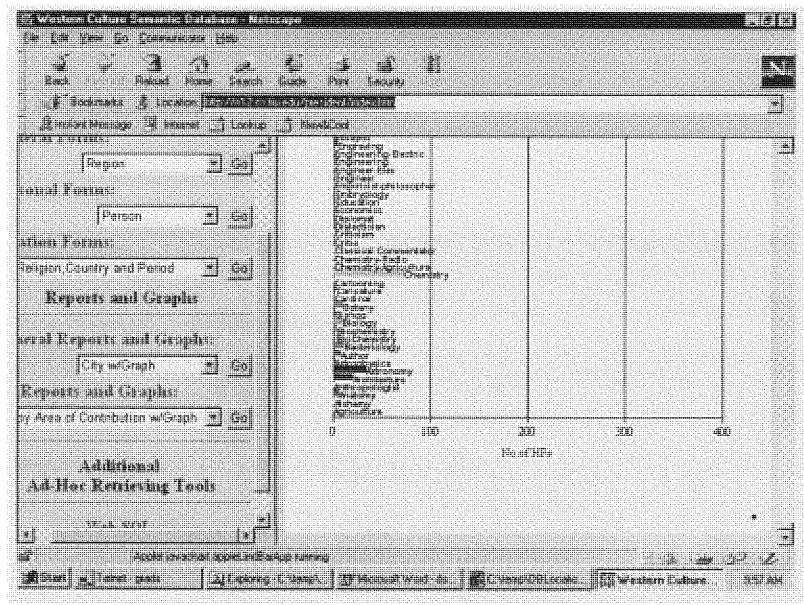


Figure 4.74 ...continuation HF by Area of Contribution – Graph for all the Areas

- HF by Nobel Prize w/Graph: This option generates a Report of all the Historical Figures in the database grouped Nobel Prize acquisition. The user can select some prize fields to display, or include all the fields in the database.

It also generates a graph of the data for each option. See **Figure 4.75** to **Figure 4.77**.

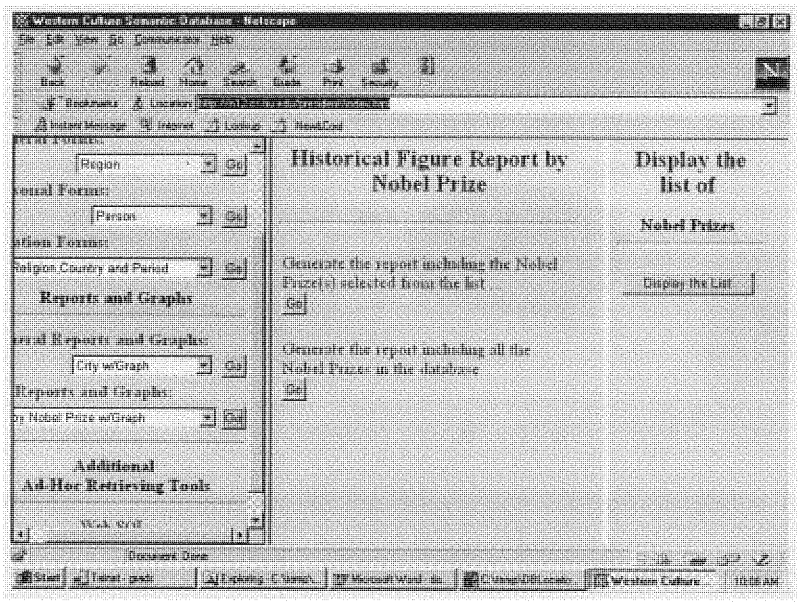


Figure 4.75 HF by Nobel Prize – Report for selected Fields

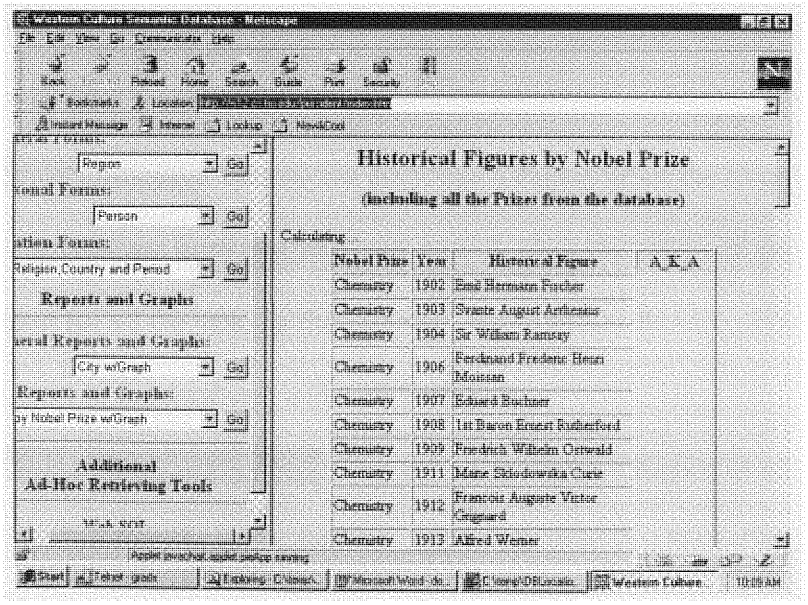


Figure 4.76 HF by Nobel Prize – Report for all the Fields in the database.

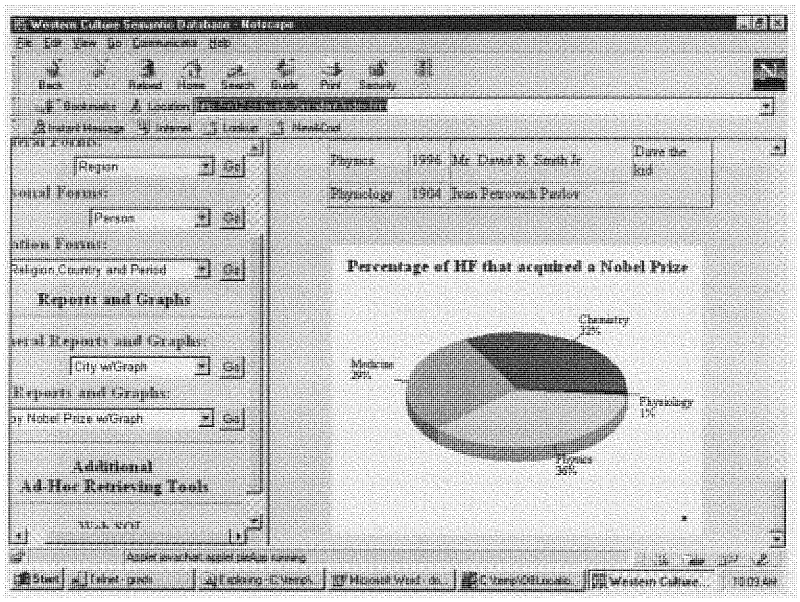


Figure 4.77 ...continuation HF by Nobel Prize – Graph for all the Prizes

- HF by Title: This option generates a Report of all the Historical Figures in the database grouped by their title of nobility. It gives the user the option of generating the report for all the titles, or only a group of them specified by a list of titles separated by commas.
- HF by Birth Year: This option generates a Report of all the Historical Figures in the database born in a particular year. It gives the option of generating the report for all the years in the database, which means that list al the Historical Figures in the database, and their birth year.
- HF by Time Span w/Graph: This option generates a Report of all the Historical Figures in the database that where alive in a Period inserted. It also generates a graph of the data. The idea of this graph is to recognize important historical facts during time, because of the peaks in the curve (when a peak

occurs, is because the number of Figures increases during that period). See from **Figure 4.78** to **Figure 4.80**.

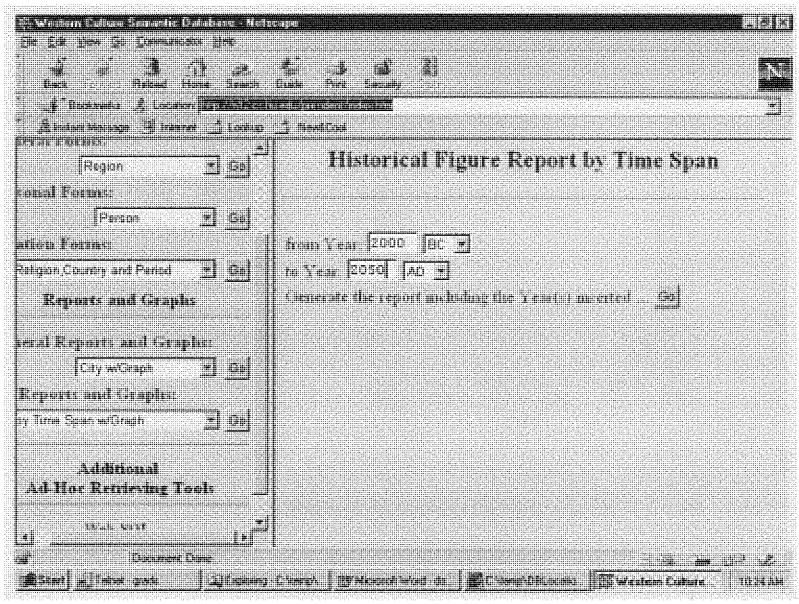


Figure 4.78 HF by Time Span – HF Reports and Graphs

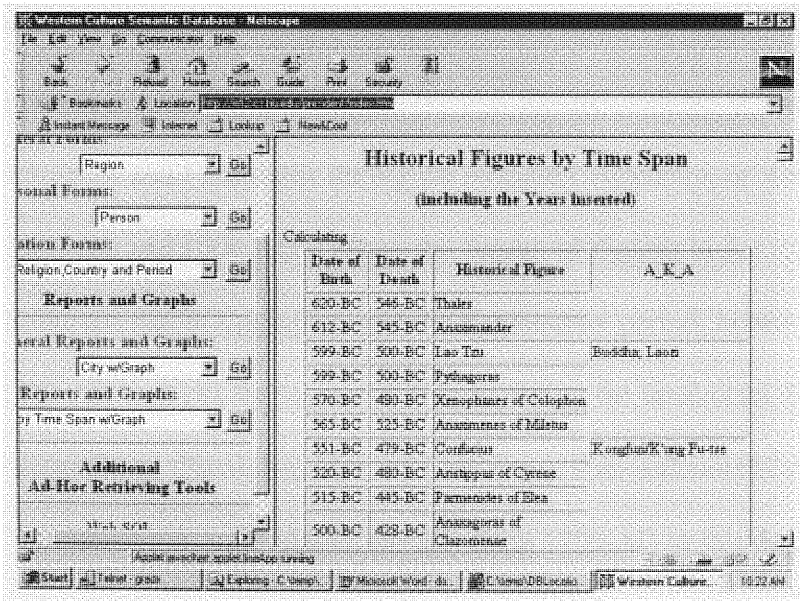


Figure 4.79 HF by Time Span – Report for the Period.

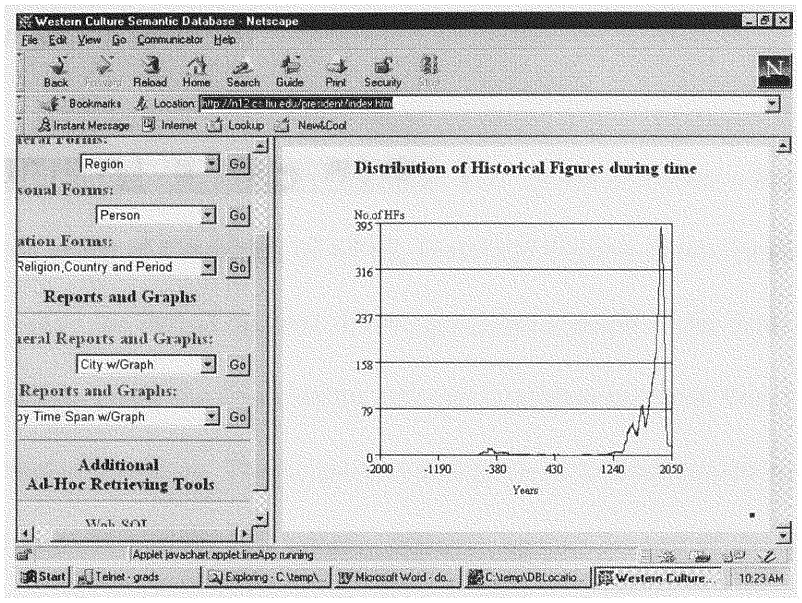


Figure 4.80 ...continuation HF by Time Span – Graph for the Period

Figure 4.80 shows important information about the impact of Historical Figures in the Western Cultures evolution. Every peak in this figure corresponds to an important period in history: between 600-BC and 400-BC it shows the impact of Figures in Science and Philosophy during the Ancient Civilization of Greece; between 1300-AD and 1500-AD it shows the impact of Figures in Art and Science during the rise of Capitalism and the Renaissance period; between 1600-AD and 1700-AD it shows the impact of Figures during the Intelectual Revolution period: Mercantilism, Classicism (Art and Literature), Rationalism(Philosophy) and the Commonwealth (in England); finnally from 1700-AD to this days it shows an important peak that start to decline around 1880-AD, showing the impact of Figures during the Agricultural Revolution, the Romanticism, the French

Revolution, the Napoleonic empire, the Industrial Revolution, and the New Capitalism period. [Burn-74]

- HF by Country and Time Span w/Graph: This option generates a Report for a selected group of Countries, of all the Historical Figures in the database that where alive the Period inserted. It also generates a graph of the data, plotting in a red line graph the first country selected, and then in a blue line, the rest of the selected countries. The purpose of this graph is to compare the impact of the Figures born in the first country selected, against the rest of the countries in the list. See from **Figure 4.81** to **Figure 4.83**.

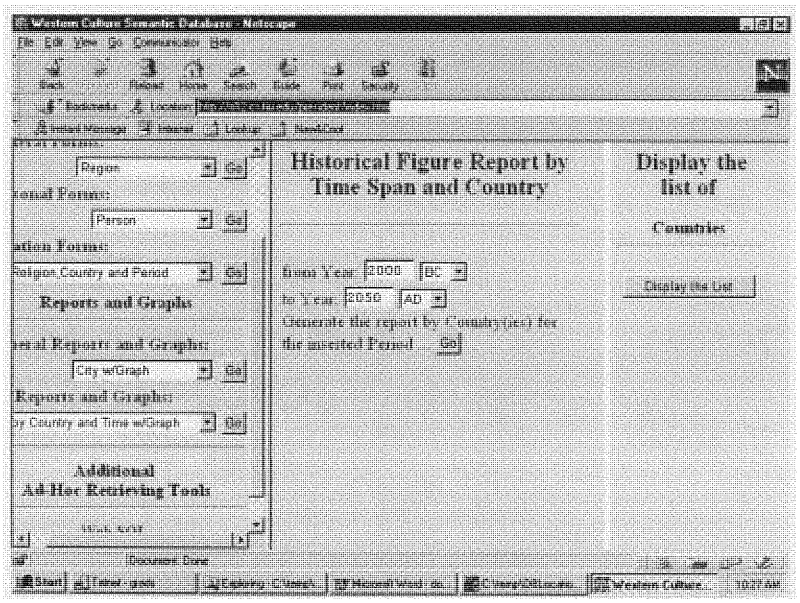


Figure 4.81 HF by Country and Time Span – HF Reports and Graphs

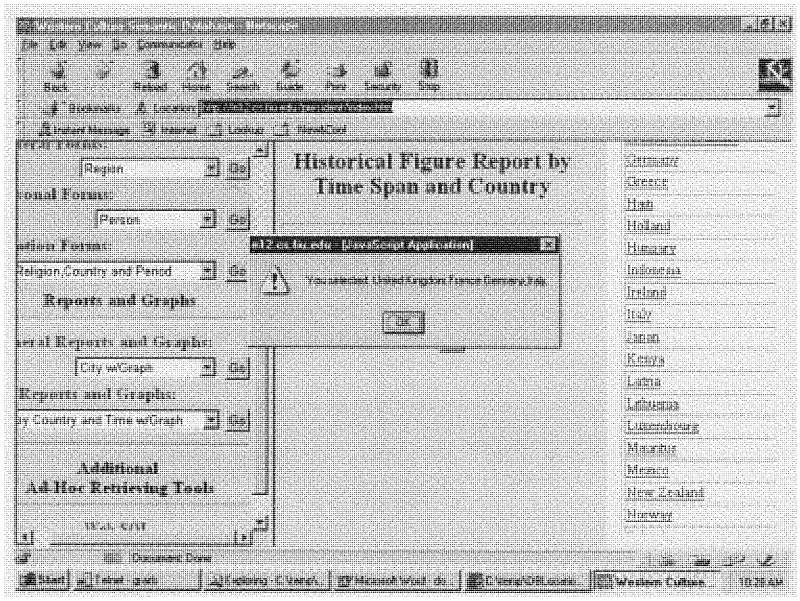


Figure 4.82 HF by Country and Time Span – Report for selected Countries

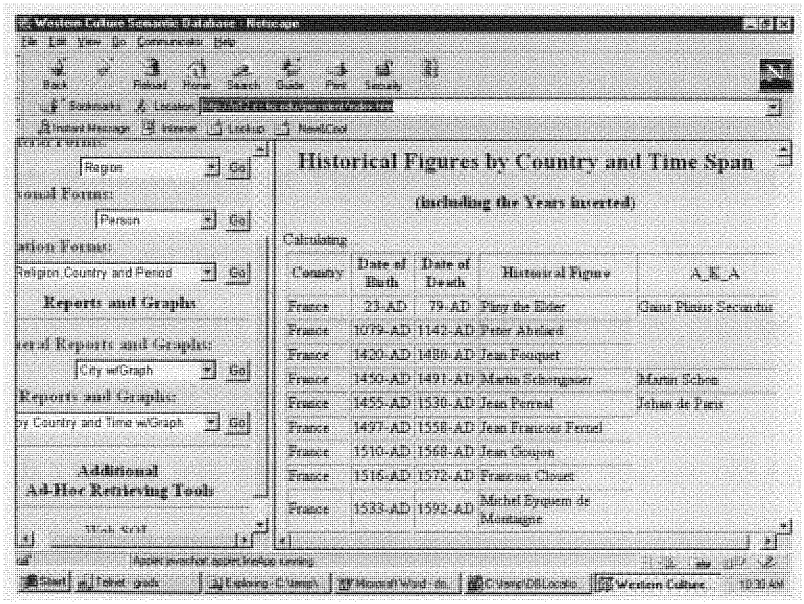


Figure 4.83 HF by Country and Time Span –

Report for selected Countries alive in the Period

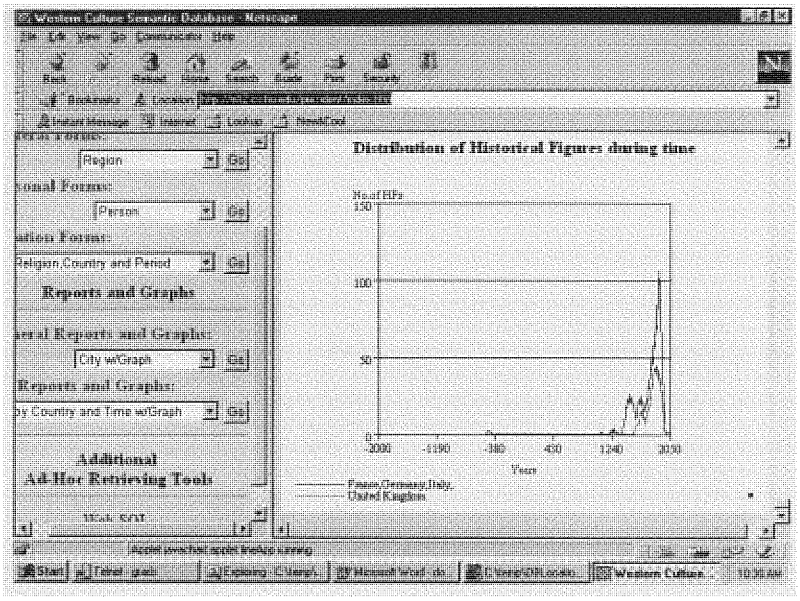


Figure 4.84 ...continuation HF by Country and Time Span – Graph for selected Countries alive in the Period

3. Advanced Ad Hoc Retrieving Tools: Includes applications to perform queries on the Western Culture Semantic Database. The user can use the default values or enter their own values and design any query to retrieve needed information. They are very powerful tools that give the user freedom to manage the data in the database according to non-predefined forms and queries.

- Web-SQL: Is a tool for web interaction at SQL level. The user can edit a query, process it, and retrieve results writing SQL statements based on the schema, provided as a guideline and against which to pose queries[11]. See **Figure 4.85** and **Figure 4.86**.

[11] This tools requires SQL knowledge.

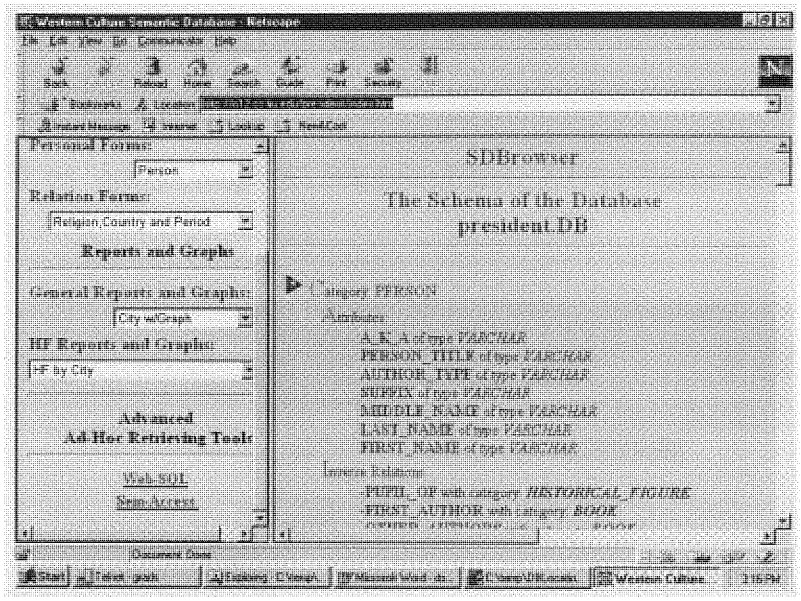


Figure 4.85 Web-SQL – Advanced Ad Hoc Retrieving Tools

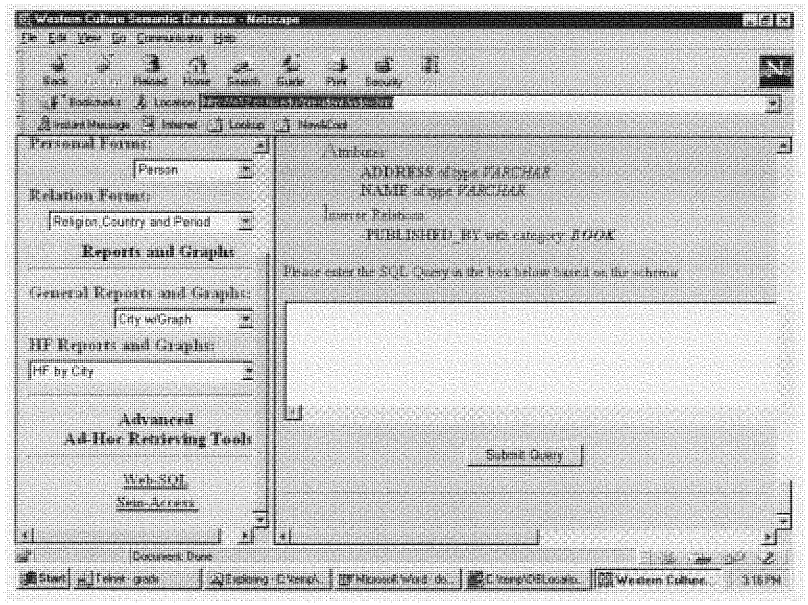


Figure 4.86 …continuation Web-SQL – Advanced Ad Hoc Retrieving Tools

Sem-Access: Auto Generator for standard Web forms and Reports. Allows the end-user to have automatically generated forms and standard or customizable reports derived from the conceptual schema of Sem-ODB. It displays all the categories of

the database and further actions are taken by clicking on the buttons. By clicking on the "Query in Progress", the user will navigate the schema. To retrieve the results, just click on "query Complete". See from **Figure 4.87** to **Figure 4.89**.

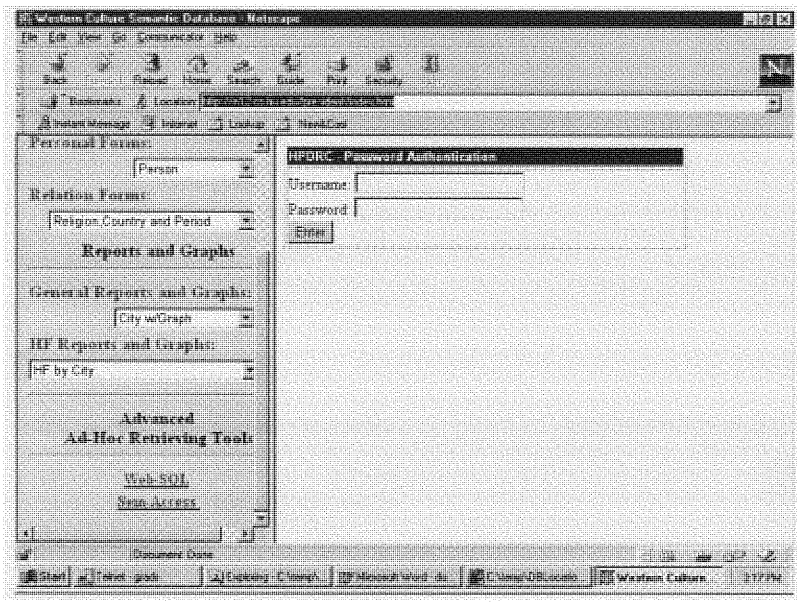


Figure 4.87 Sem-Access, security window– Advanced Ad Hoc Retrieving Tools

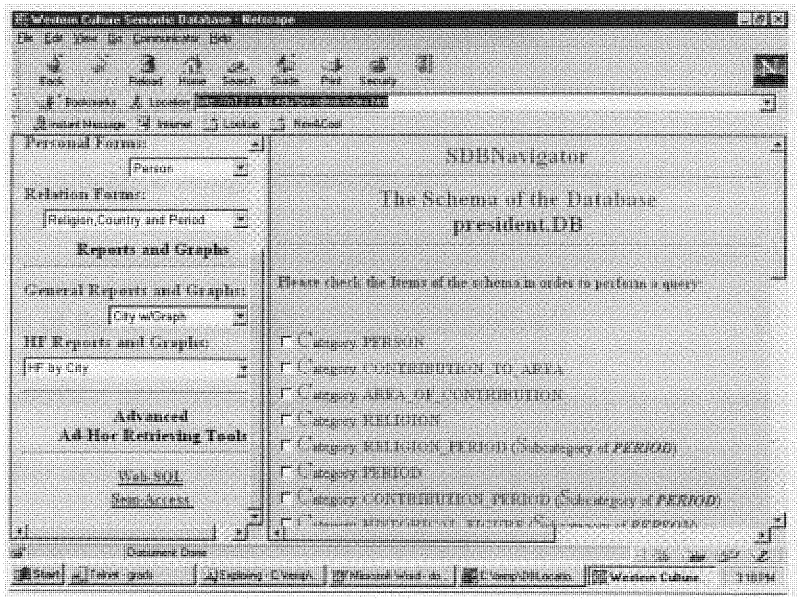


Figure 4.88 Sem-Access – Advanced Ad Hoc Retrieving Tools

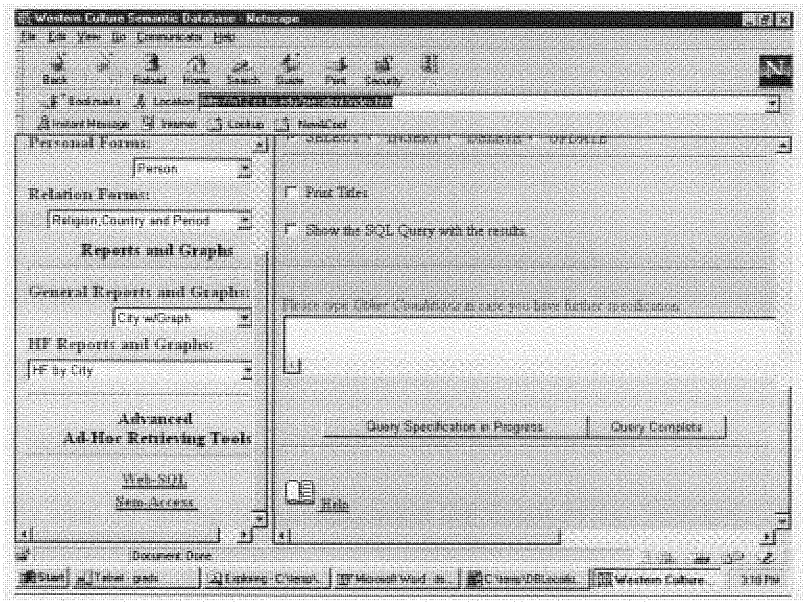


Figure 4.89 …continuation Sem-Access – Advanced Ad Hoc Retrieving Tools

Advantages of a Semantic Implementation approach

One purpose of this thesis is to demonstrate that using a Semantic methodology for developing the application brings advantages over solving the problem with a Relational methodology.

The advantages of the Semantic application are as follows:

- Web interface application that can be used for research by any interested person.
- Friendlier and more intelligent user interface based on the stored meaning of the data, comprehensive enforcement of integrity constraints and greater flexibility.
- Data items related to objects can be of arbitrary size, multi-valued, or missing entirely. Remember that in this project a lot of data has not yet been collected.
- In particular, the use of semantics allows distributing data among different processors in a way, which is transparent to both database programmers and database users.
- Efficient and semantically safe concurrency control, using the algorithms developed at HPDRC.

As mentioned before, in theory a Semantic model has advantages, but also in practice we can support this idea by taking in consideration the result of the Semantic Implementation approach: the "Western Cultures Semantic Database application" compared with the partial result of the Relational approach: the Data Entry Forms.

V. CONCLUSION

The Relational database approach has provided a good result in many conventional database applications, being the most popular technique commercially speaking. However, in some situations other approaches need to be considered, and the Western Culture Database Project is a clear example where a Semantic (and/or Object-Oriented) database was necessary to achieve the main result of this project: **The "Western Cultures Semantic Database application" - web interface.**

As a result of the study, it can be concluded, that using a Semantic methodology for developing the application brings advantages over solving the problem with a Relational methodology. These work shows that all the capabilities of a Relational approach can be implemented using the Semantic/Object Oriented tools available at HPDRC, and requires less effort and time for development and future maintenance.

The resulting semantic web interface, permits the efficient storage and retrieval of data, about persons considered to be historical due to their contributions in a specific field. It also provids also a useful graphical display of the data generating important information to support research assertions about the impact of historical facts in the evolution of western cultures.

VI. FUTURE WORK

Some additional aspects can be included to this project. The first point to mention is a more complete data recollection, by the President's office, about Historical Figures in fields like literature, music, etc. Also some data about regions, religions, population, periods, books and references, is missing to complete the database.

An important aspect to add to the web interface implementation, is an option for citations (books and references), pending at this time, because of the necessity of its revision by the user. The design of this section is included, and the Semantic database also covers it, but probably the current design will change, and the proposed design (sub-schema presented as an Appendix) is the one that will be implemented.

Finally, in the web interface, the inclusion of more graphics and multi-media data (for example, pictures or sound files of Historical Figures) will make the application more complete.

LIST OF REFERENCES

BOOKS

[Atre-80] Atre, S. *Data Base: Structured Techniques for Design, Performance, and Management.* A Wiley-Interscience Publications, 1980.

[Burn-74] Burns, R. *World Civilizations.* W.W. Norton & Company Inc., 1974.

[Card&McLeod-90] Cárdenas, A.F., and McLeod D., *Research Foundations in Object-Oriented and Semantic Database Systems.* Prentice Hall, 1990.

[Codd-70] Codd, E.F. *A Relational Model for Large Shared Data Banks* Comm. of ACM, Vol. 13, No. 6, June 1970.

[Codd-82] Codd, E.F. *Relational Databases: A Practical Foundation of Productivity* Comm. of ACM, Vol. 25, No. 2, February 1982.

[Darn-98] Darnell, R. *HTML 4 Professional Reference Edition.* Macmillan Computer Publishing, 1998.

[Date-86] Date, C.J. *Relational Database: Selected Writings.* Addison-Wesley Publishing Company, 1986.

[Good-98] Goodman, D. *JavaScript Bible.* IDG Books Worldwide, Inc., 1998.

[Gupta&Horo-91] Gupta, R. and Horowitz, E. *Object-Oriented Databases with Applications to Case, Networks, and VLSI CAD.* Prentice Hall, 1991.

[Koch&Loney-97] Koch, G. and Loney, K. *Oracle: The Complete Reference.* Electronic Edition, Oracle Press, 1997.

[Rishe-92] Rishe, N. *Database Design: The Semantic Modeling Approach.* McGraw Hill, 1992.

[Stevens-94] Stevens, Al. *C++ Database Development.* 2nd. Edition, MIS Press, 1994.

[Siyan&Weaver-98] Siyan, K and Weaver J. *Inside Java.* New Riders Publishing, 1998.

[Ullm&Wido-97] Ullman, J.D. and Widom, J. *A First Course in Database Systems.* Prentice Hall, 1997.

[Vald&Garad-89] Valduriez, P. and Garadin, G. *Analysis and Comparison of Relational Database Systems.* Addison-Wesley Publishing Company, 1989.

[Yao-85] Yao, B. *Principles of Database Design*. Prentice Hall, 1985.

PAPERS and WEB SITES

[JavaChart-98]. *JavaChart Applets and Class Library,* http://www.ve.com/javachart/

[Rishe-98] Rishe, N. *Semantic, Relational and Object-Oriented Databases*, http://hpdrc.cs.fiu.edu/library/papers/SDBMS.RDBMS.OODBMS.html

[Rishe-93] Rishe, N. *A Methodology and Tool for Top-down Relational Database Design*, http://hpdrc.cs.fiu.edu/docs/papers/Rishe-93-MT.txt, 1993

[Rishe-98] Rishe, N. *Semantic SQL*, 1998

[Sem-ODB-98] HPDRC. *Semantic Object-oriented Database Management System, C++ interface, Programmer's guide*, 1998.

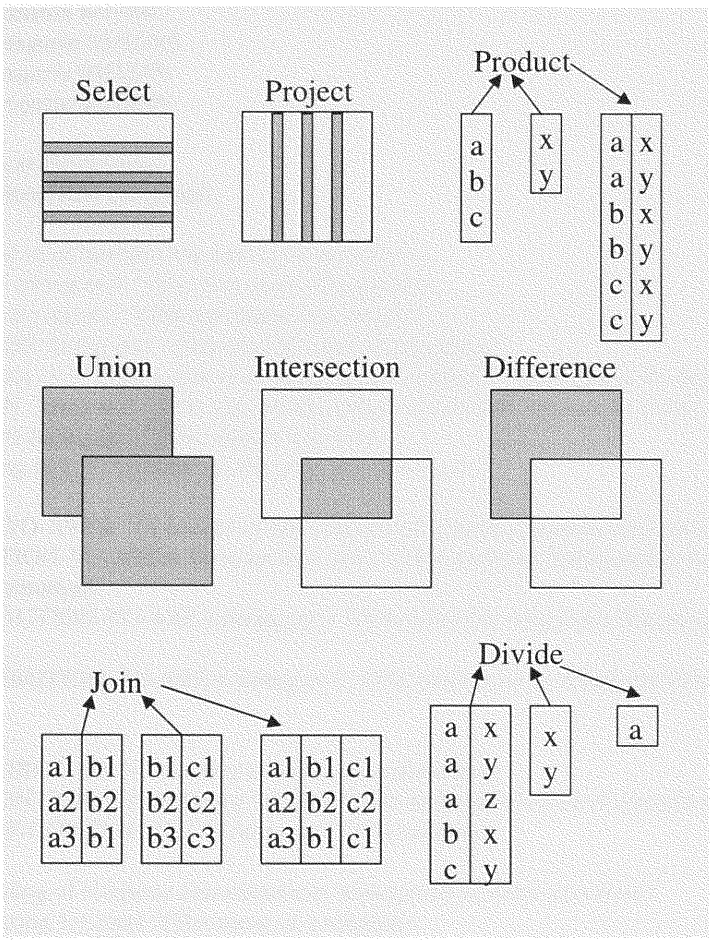
[Nabou-98] Naboulsi, K. *Structured Query Language (SQL),* http://www.cs.fiu.edu/~knabou01/intro.html

[WebRG-98] Naboulsi, K. *WWW Database Application Interface Tool*, http://n38.cs.fiu.edu/DemoInfo.nsf/ WebRG project, 1998

APPENDICES

APPENDIX A (theory appendix)

The relational algebra [Date-86]



APPENDIX B (design appendix)

".sdl" File for the Semantic Schema

```
|| Historical Figures
.tC author Rosy--Rodriguez 07/09/96
.tC client Mayra--Beers
.tC change Naphtali--Rishe 7/11/96 8/8/96 8/12/96
.tC change Sergey--Fedorishin 8/17/96
.tC change Raquel--Ghersgorin 03/10/98
.tC change Raquel-Ghersgorin 03/31/98
.tC change Raquel-Ghersgorin 06/29/98

.rC CITY WC-SubSchema-Geography
.rC COUNTRY WC-SubSchema-Geography

.cA PERSON "(A record of all persons relevant to this database)
.aT first-name PERSON String m:1 "(The first name of a person)
.aT last-name PERSON String m:1 "(The last name of a person)
.aT middle-name PERSON String m:1 "(The middle name of a person.)
.aT suffix PERSON String m:1 "(The suffix of a name, e.g. Jr in Martin Luther King Jr., II in Elizabeth II.)
.aT author-type PERSON String m:1 "(The type of the person when is an author of a book)
.aT person-title PERSON String m:1 "(The title of a person)
.aT a-k-a PERSON String m:1 "(The name as the person is known as )

.cA CONTRIBUTION-TO-AREA "(A record of associations of historical figures with specific areas)
.aT major CONTRIBUTION-TO-AREA Boolean m:1,total "(This attribute distinguishes a contribution
considered a major contribution)
.aT school-of CONTRIBUTION-TO-AREA String m:1 "(The school to where the Historical Figure made
the contribution)
.aT profession CONTRIBUTION-TO-AREA String m:1 "(The profession to which the Historical Figure
made the contribution)

.cA AREA-OF-CONTRIBUTION "(A catalog of areas of contribution)
.aT name AREA-OF-CONTRIBUTION String 1:1,total "(The name of the area of contribution)
.aT comments AREA-OF-CONTRIBUTION String m:1 "(A comment)

.cA RELIGION "(A catalog of religions denominations widespread in some countries)
.aT name RELIGION String 1:1,total "(The name of a religion)

.cA RELIGION-PERIOD "(A record of religions of particular countries during specific time periods. A
country can have several widespread religions at the same time. In addition, countries change their religion
over time.)

.cA PERIOD "(A period of time)
.aT from-year PERIOD -2000..2050 m:1 "(The year of period beginning.)
.aT to-year PERIOD -2000..2050 m:1 "(The year of period end.)
.aT percentage PERIOD 1..100 m:1 "(An estimate of percentage of the population of the country which
practiced the religion or contributed on a specific area during a particular period)

.cA CONTRIBUTION-PERIOD "(A record of contribution of a particular historical person to specific area
during time periods.)
```

.cA HISTORICAL-FIGURE "(Historical figures are persons who are subjects of this study.)
.aT date-of-birth HISTORICAL-FIGURE -2000..2050 m:1 "(The date of birth, if known)
.aT date-of-death HISTORICAL-FIGURE -2000..2050 m:1 "(The date of death, if known)
.aT birth-order HISTORICAL-FIGURE Number m:1 "(A generation which the historical figure belongs to)
.aT gender HISTORICAL-FIGURE 'm','f' m:1 "(The gender of the person , i.e. m:male, f:female)"
.aT parent HISTORICAL-FIGURE Boolean m:1 "(The parent's occupation. If it is true it means that the person's father did the same work as the person)"
.aT funding HISTORICAL-FIGURE 'independent','patron','institution','other' m:1 "(The type of funding the person received)
.aT comments HISTORICAL-FIGURE String m:1 "(Comment for Historical Figure. May be used also to indicate that the country of birth is different from country of work)
.aT number-of-references HISTORICAL-FIGURE Integer m:1 "(The number of references of that Historical Figure in the literature)
.aT method HISTORICAL-FIGURE Integer m:1 "(The method's number used to collect the information about an Historical Figure)

.sC RELIGION-PERIOD PERIOD
.sC CONTRIBUTION-PERIOD PERIOD
.sC HISTORICAL-FIGURE PERSON

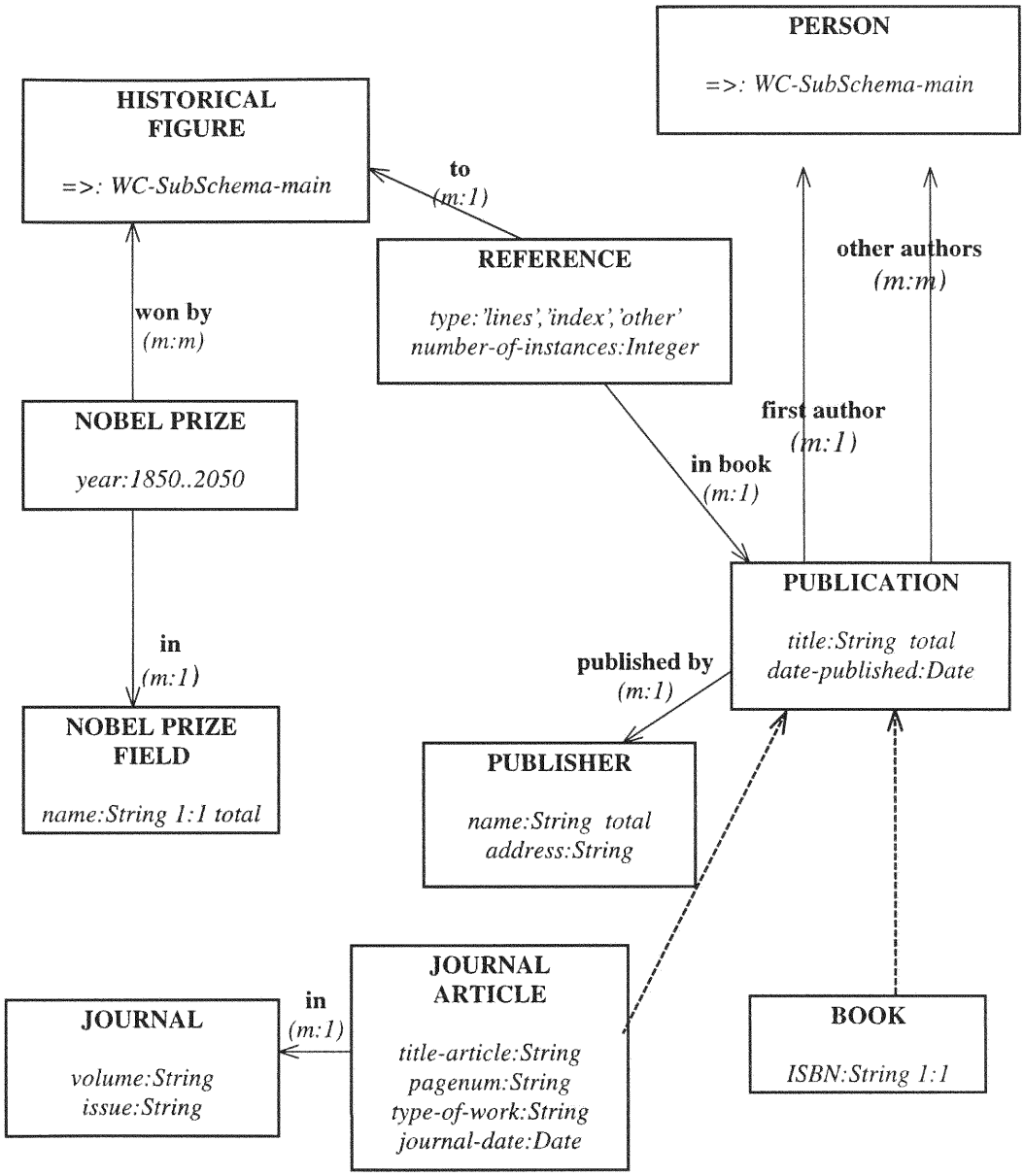
.rE of CONTRIBUTION-TO-AREA AREA-OF-CONTRIBUTION m:1 "(The area to which a contribution was made)
.rE by CONTRIBUTION-TO-AREA HISTORICAL-FIGURE m:1 "(The historical figure by whom the contribution was made)
.rE during-period CONTRIBUTION-TO-AREA CONTRIBUTION-PERIOD m:1 "(The period during which a contribution was made)
.rE is-part-of AREA-OF-CONTRIBUTION AREA-OF-CONTRIBUTION m:1 "(The area of which this is a subarea)
.rE of-religion RELIGION-PERIOD RELIGION m:1 "(The subject religion during a particular period)
.rE of-country RELIGION-PERIOD COUNTRY m:1 "(The subject country)
.rE has-practiced HISTORICAL-FIGURE RELIGION m:m "(The religion practiced by a historical figure. A person may have practiced several religions)
.rE worked-in HISTORICAL-FIGURE CITY m:1 "(The modern country within which modern borders the person was working)
.rE born-in HISTORICAL-FIGURE CITY m:1 "(The modern country within which modern borders the person was born)
.rE born-in-country HISTORICAL-FIGURE COUNTRY m:1 "(The country where the historical figure was born)
.rE worked-in-country HISTORICAL-FIGURE COUNTRY m:1 "(The country were the Historical Figure worked in)
.rE pupil-of HISTORICAL-FIGURE PERSON m:m "(The teacher of that Histrocial Figure)
.tC abs PERSON 2.68571 3.65
.tC abs CONTRIBUTION-TO-AREA 2.81429 0.991667
.tC abs AREA-OF-CONTRIBUTION 2.75 1.95833
.tC abs RELIGION 1.72143 1.13333
.tC abs RELIGION-PERIOD 0.671429 0.00833333
.tC abs PERIOD 1.63571 0.00833333
.tC abs CONTRIBUTION-PERIOD 2.82857 0
.tC abs HISTORICAL-FIGURE 0 3.95
.tC relcorn of CONTRIBUTION-TO-AREA n AREA-OF-CONTRIBUTION s
.tC put of CONTRIBUTION-TO-AREA AREA-OF-CONTRIBUTION rjust
.tC relcorn by CONTRIBUTION-TO-AREA nw HISTORICAL-FIGURE se
.tC put by CONTRIBUTION-TO-AREA HISTORICAL-FIGURE ljust

```
.tC put during-period CONTRIBUTION-TO-AREA CONTRIBUTION-PERIOD rjust
.tC relcorn of-religion RELIGION-PERIOD ne RELIGION w
.tC put of-religion RELIGION-PERIOD RELIGION ljust
.tC relcorn of-country RELIGION-PERIOD n COUNTRY s
.tC put of-country RELIGION-PERIOD COUNTRY ljust
.tC relcorn has-practiced HISTORICAL-FIGURE se RELIGION n
.tC put has-practiced HISTORICAL-FIGURE RELIGION rjust
.tC relcorn worked-in HISTORICAL-FIGURE e CITY w
.tC relcorn born-in HISTORICAL-FIGURE se CITY sw
.tC relcorn born-in-country HISTORICAL-FIGURE sw COUNTRY n
.tC put born-in-country HISTORICAL-FIGURE COUNTRY ljust
.tC relcorn worked-in-country HISTORICAL-FIGURE s COUNTRY ne
.tC put worked-in-country HISTORICAL-FIGURE COUNTRY ljust
.tC relcorn pupil-of HISTORICAL-FIGURE ne PERSON w
.tC subcorn RELIGION-PERIOD e PERIOD w
.tC subcorn CONTRIBUTION-PERIOD w PERIOD e
.tC subcorn HISTORICAL-FIGURE e PERSON w
.pS Historical Figures, Semantic main schema
.mR WC-SubSchema-main

|| Historical Figures
.tC author Rosy--Rodriguez 07/09/96
.tC client Mayra--Beers
.tC change Raquel--Ghersgorin 03/31/98
.tC change Raquel-Ghersgorin 06/29/98

.cA REGION "(A region, sub-region, continent, or any other geographical area which may be of interest)
.aT name REGION String m:1 "(A name of the Region)

.cA WORLD-POPULATION

.cA POPULATION "(A record of censuses or estimates of population of countries and the world)
.aT number-in-millions POPULATION Number m:1,total "(The population)
.aT year POPULATION -2000..2050 m:1,total "(The year of census or estimate of the population)

.cA CITY "(The record of all cities relevant to this study)
.aT name CITY String m:1

.cA COUNTRY "(Table of countries relevant to this study)
.aT name COUNTRY String 1:1,total "(The modern name of the Country)
.aT symbol COUNTRY Char(4) 1:1,total "(The international 4-letter symbol of the country, e.g.
US,RU,AU,CA,USVI)
.aT western-since COUNTRY -2000..2050 m:1 "(The beginning year in which the country was considered
part of Western Civilization)
.aT western-until COUNTRY -2000..2050 m:1 "(The beginning year in which the country was considered
part of Western Civilization)

.sC WORLD-POPULATION POPULATION

.rE is-part-of REGION REGION m:1
.rE belongs-to CITY COUNTRY m:1
.rE had-population COUNTRY POPULATION 1:m "(Population record of the Country)
.rE is-in COUNTRY REGION m:m
```

```
.tC abs REGION 1.11429 0.166667
.tC abs WORLD-POPULATION 2.87857 3.325
.tC abs POPULATION 0.285714 3.16667
.tC abs CITY 2.82857 2.65833
.tC abs COUNTRY 0.464286 1.69167
.tC put belongs-to CITY COUNTRY ljust
.tC relcorn had-population COUNTRY n POPULATION s
.tC relcorn is-in COUNTRY s REGION n
.tC subcorn WORLD-POPULATION w POPULATION e
.pS Historical Figures - geographical semantic sub-schema
.mR WC-SubSchema-Geography

|| Historical Figures
.tC author Rosy--Rodriguez 07/09/96
.tC client Mayra--Beers
.tC change Raquel--Ghersgorin 03/31/98
.tC change Raquel-Ghersgorin 06/29/98

.rC PERSON WC-SubSchema-main
.rC HISTORICAL-FIGURE WC-SubSchema-main

.cA NOBEL-PRIZE "(A catalog of Nobel Prizes)
.aT year NOBEL-PRIZE 1850..2050 m:1 "(The year in which a Nobel Prize was awarded to a particular
person)

.cA NOBEL-PRIZE-FIELD "(A catalog of fields in which Nobel Prizes are awarded)
.aT name NOBEL-PRIZE-FIELD String 1:1,total "(The name of the field of the Nobel Prize)

.cA REFERENCE "(A catalog of references to persons in books)
.aT type REFERENCE 'lines','index','other' m:1 "(The type of reference made. e.g. by counting lines, by
looking in the index)
.aT number-of-instances REFERENCE Integer m:1 "(The number of references to a particular historical
figure found in the book)

.cA PUBLISHER "(A catalog of publishers)
.aT name PUBLISHER String m:1,total "(The name of the publisher)
.aT address PUBLISHER String m:1 "(The address of the publisher)

.cA BOOK "(A catalog of books in which references were located)
.aT ISBN BOOK String 1:1 "(The International Standard Book Number, if known)
.aT title BOOK String m:1,total "(The title of the book)
.aT date-published BOOK Date m:1 "(The date of publication, if known)

.rE in NOBEL-PRIZE NOBEL-PRIZE-FIELD m:1 "(The field in which the Nobel Prize was won)
.rE won-by NOBEL-PRIZE HISTORICAL-FIGURE m:m "(The historical figure who won the Nobel Prize)
.rE to REFERENCE HISTORICAL-FIGURE m:1 "(The historical figure who is referenced)
.rE in REFERENCE BOOK m:1 "(The book in which the reference was found.)
.rE published-by BOOK PUBLISHER m:1 "(The publisher of the book, if known)
.rE first-author BOOK PERSON m:1 "(The main person who wrote the book)
.rE other-authors BOOK PERSON m:m "(The co-authors of the book, other than the first author.)
.tC abs NOBEL-PRIZE 0.6 2.425
.tC abs NOBEL-PRIZE-FIELD 0.535714 1.4
.tC abs REFERENCE 1.5 2.85833
```

```
.tC abs PUBLISHER 3.19286 1.41667
.tC abs PUBLICATION 1.86429 1.49167
.tC abs BOOK 2.76429 0.391667
.tC relcorn in NOBEL-PRIZE s NOBEL-PRIZE-FIELD n
.tC put in NOBEL-PRIZE NOBEL-PRIZE-FIELD rjust
.tC relcorn won-by NOBEL-PRIZE n HISTORICAL-FIGURE s
.tC relcorn to REFERENCE n HISTORICAL-FIGURE se
.tC put to REFERENCE HISTORICAL-FIGURE ljust
.pS Historical Figures- Prizes and Citations semantic sub-schema
.mR WC-SubSchema-Prizes-and-Citations
```

Purposed Prizes and Citations semantic sub-schema



☐ *PERSON* — category (See subschema WC-SubSchema-main.)

☐ *HISTORICAL-FIGURE* — category (See subschema WC-SubSchema-main.)

☐ *NOBEL-PRIZE* — category (A catalog of Nobel Prizes)

- ☐ *year* — attribute of *NOBEL-PRIZE*, range: *1850..2050* (*m:1*) (The year in which a Nobel Prize was awarded to a particular person)

- ☐ *NOBEL-PRIZE-FIELD* — category (A catalog of fields in which Nobel Prizes are awarded)

- ☐ *name* — attribute of *NOBEL-PRIZE-FIELD*, range: *String* (*1:1,total*) (The name of the field of the Nobel Prize)

- ☐ *REFERENCE* — category (A catalog of references to persons in books)

- ☐ *type* — attribute of *REFERENCE*, range: *'lines', 'index', 'other'* (*m:1*) (The type of reference made. e.g. by counting lines, by looking in the index)

- ☐ *number-of-instances* — attribute of *REFERENCE*, range: *Integer* (*m:1*) (The number of references to a particular historical figure found in the book)

- ☐ *JOURNAL-ARTICLE* — category (A journal article in which references were located)

- ☐ *title-article* — attribute of *JOURNAL-ARTICLE*, range: *String* (*m:1*) (The title of the article where the reference was found)

- ☐ *pagenum* — attribute of *JOURNAL-ARTICLE*, range: *String* (*m:1*) (The number of pages referencing the Historical Figure)

- ☐ *type-of-work* — attribute of *JOURNAL-ARTICLE*, range: *String* (*m:1*)

- ☐ *journal-date* — attribute of *JOURNAL-ARTICLE*, range: *Date* (*m:1*)

- ☐ *PUBLISHER* — category (A catalog of publishers)

- ☐ *name* — attribute of *PUBLISHER*, range: *String* (*m:1,total*) (The name of the publisher)

- ☐ *address* — attribute of *PUBLISHER*, range: *String* (*m:1*) (The address of the publisher)

- ☐ *PUBLICATION* — category (A catalog of publications in which references were located)

- ☐ *title* — attribute of *PUBLICATION*, range: *String* (*m:1,total*) (The title of the book or journal article)

- ☐ *date-published* — attribute of *PUBLICATION*, range: *Date* (*m:1*) (The date of publication, if known)

- ☐ *BOOK* — category (A catalog of books in which references were located)

- ☐ *ISBN* — attribute of *BOOK*, range: *String* (*1:1*) (The International Standard Book Number, if known)

- ☐ *JOURNAL* — category (A catalog of journals in which references were located)

- ☐ *volume* — attribute of *JOURNAL*, range: *String* (*m:1*) (The volume of the journal where the reference was found)

- ☐ *issue* — attribute of *JOURNAL*, range: *String* (*m:1*) (The issue of the journal where the reference was found)

- ☐ *in* — relation from *NOBEL-PRIZE* to *NOBEL-PRIZE-FIELD* (*m:1*) (The field in which the Nobel Prize was won)

- ☐ *won-by* — relation from *NOBEL-PRIZE* to *HISTORICAL-FIGURE* (*m:m*) (The historical figure who won the Nobel Prize)

- ☐ *to* — relation from *REFERENCE* to *HISTORICAL-FIGURE* (*m:1*) (The historical figure who is referenced)

- ☐ *in* — relation from *REFERENCE* to *PUBLICATION* (*m:1*) (The publication in which the reference was found.)

- ☐ *in* — relation from *JOURNAL-ARTICLE* to *JOURNAL* (*m:1*)

- ☐ *published-by* — relation from *PUBLICATION* to *PUBLISHER* (*m:1*) (The publisher of the publication, if known)

- ☐ *first-author* — relation from *PUBLICATION* to *PERSON* (*m:1*) (The main person who wrote the publication)

- ☐ *other-authors* — relation from *PUBLICATION* to *PERSON* (*m:m*) (The co-authors of the publication, other than the first author.)

Snapshot of Artists File

Microsoft Excel - !art-sho.rtf

Last Na	First Nan	Middle N	Suffix	Title	A.K.A	Inde	Hits	Birth	Birth	DOB	DOD	Primary	Secon	School c	School	Com
Avercam	Hendrick					2	6	Ams	HD	1585-AD	1634-AD	Painting				
Barye	Antoine-Louis					8	6	Lyon	FR	1796-AD	1875-AD	Sculpture	Painting			
Baugin	Lubin					1	6		FR	1610-AD	1663-AD	Painting				
Bloemae	Abraham					2	6		HD	1564-AD	1651-AD	Painting				
Bosse	Abraham					8	6		FR	1602-AD	1676-Ad	Engraving				
Bouchar	Edme					2	6		FR	1698-AD	1762-AD	Sculpture		Baroque		
Bril	Paul					2	6	Antw	BE	1554-AD	1626-AD	Painting				
Burgkma	Hans		the Elder			3	6		GN	1473-AD	1531-AD	Painting	Woodcutting			
Cappelle	Jan	van de				1	6		HD	1624-AD	1679-AD	Painting	Engraving			
Carlevari	Luca					3	6	Udin	IT	1665-AD	1731-AD	Painting				
Carpeau	Jean-Baptiste					6	6		FR	1827-AD	1875-AD	Sculpture				
Castiglio	Giovanni	Benedetto			Il Grec	3	6	Genc	IT	1610-AD	1665-AD	Painting	Etching			
Chasseri	Theodore					6	6	Antill	FWI	1819-AD	1856-AD	Painting				
Cooper	Samuel					3	6		UK	1609-AD	1660-AD	Painting		Baroque		
Cossa	Francesc	del				3	6		IT	1430-AD	1477-AD	Painting				
Crespi	Giuseppe	Maria			Lo Sp	3	6	Bolo	IT	1665-AD	1747-AD	Painting				
David	Gerard					5	6	Oude	HD	1460-AD	1523-AD	Painting				c.-birt
Desiderio	da Settignano					2	6	Setti	IT	1428-AD	1461-AD	Sculpture				c.-birt
Fetti	Domenico					2	6		IT	1589-AD	1624-AD	Painting				
Feurbac	Anselm					14	6		GN	1829-AD	1880-AD	Painting		Neoclassical		
Foppa	Vincenzo					5	6	Bres	IT	1427-AD	1515-AD	Painting				
Girodet	Anne-Louis				Girode	6	6	Mont	FR	1767-AD	1824-AD	Painting	Illustration			

Snapshot of Science File

Microsoft Excel - !science.-sh

Last Na	First Na	Middle	Suffix	Title	A.K.	Gende	Hit-In	Birth City	Birth	Nobel Pr	NP Yea	Nobe	NP Y	Contributio	Contrib	Birth
Carringto	Richard	Christopher				m	4		UK					Astronomy		1824-
Cayley	Arthur					m	11	Richmond	UK					Mathematics		1821-
Bridgmar	Percy	Williams				m	5	Cambridge	US	Physics	1946			Physics		1882-
Werner	Alfred					m	5	Mulhouse,	FR	Chemistr	1913			Chemistry		1866-
Fernel	Jean	Francois				m	4	Clermont c	FR					Physician		1497-
Magendi	Francois					m	6	Bordeaux	FR					Physiology		1783-
Poncelet	Jean	Victor				m	7	Metz	FR					Math	Enginee	1788-
Viete	Francois				Fransm		11	Fontenay-l	FR					Mathematics		1540-
Wurtz	Charles	Adolphe				m	7	Wolfisheim	FR					Chemistry		1817-
Mitscher	Eilhardt					m	2	Neuende,	GN					Chemistry		1794-
Schwabe	Heinrich	Samuel				m	6	Dessau	GN					Astronomy		1789-
Waldeye	Heinrich	Wilhelm Gottfivon				m	2	Hehlen	GN					Anatomy	Medicin	1836-
Wieland	Heinrich	Otto				m	4	Pforzheim	GN	Chemistr	1927			Chemistry		1877-
Windaus	Adolf					m	3	Berlin	GN	Chemistr	1928			Chemistry		1876-
Eratosthenes	of Cyrene					m	17	Cyrene	LY					Polymath		276-E
Spallanz	Lazzaro					m	15	Modena	IT					Biology		1729-
Tartaglia	Niccolo					m	10	Fontana	IT					Mathematics		1500-
Lebedev	Pyotr	Nicolaievich				m	1	Moscow	RS					Physics		1866-
Gray	Stephen					m	7		UK					Physics		1666-
Haldane	John	Burdon Sanderson				m	5	Oxford	UK					Physiology		1892-
Mayow	John					m	10	Morval, Co	UK					Chemistry	Medicin	1640-
Williams	Alexand	William				m	8	London	UK					Chemistry		1824-

Snapshot of Philosophy File

Microsoft Excel - lphiloso.phy

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Last Nam	First Na	Middl	Suffi	Title	A.K.A.	Gen	Inde	City	Coun	Contribut	Contrib	Birth	Death	Comm	Hits	School :
2	Anselm				Saint		m		Aosta	IT	Theology		1033-AD	1109-AD		6	Scholas
3	Al-Ghazali						m		Khurasar	IR	Theology	Jurist	1058-AD	1111-AD		6	
4	Cicero	Marcus	Tullius			Tully	m		Arpinum	IT	Oratory	statesm	106-BC	43-BC		6	
5	Abelard	Peter					m		Nantes	FR	Theology		1079-AD	1142-AD		6	
6	Averroes					Ibn Rus	m		Cordoba	ES	Jurist	Medicin	1126-AD	1198-AD		6	
7	Chu Hsi					Zhu Hsi	m		Yu-hsi	CN	Classical	Historiar	1130-AD	1200-AD		6	Neo-Cor
8	Grossetes	Robert					m		Stradbro	UK	Bishop		1175-AD	1253-AD		6	
9	Bacon	Roger				Doctor I	m		Ilchester	UK	Science		1220-AD	1292-AD		6	
10	Bonaventure				Saint	Giovanr	m		Orvieto	IT	Theology		1221-AD	1274-AD		6	
11	Aquinas	Thomas			Saint	Doctor /	m		Aquino	IT	Theology		1225-AD	1274-AD		6	
12	Duns Scot	John				Doctor :	m		Maxton	UK	Theology		1265-AD	1308-AD		6	
13	Ockham	William				The Ver	m		Ockham	UK	Scholastic	philoso	1285-AD	1349-AD	c.-birtl	6	
14	Galennus	Claudius				Galen	m		Pergamu	TR	Medicine		130-AD	201-AD		6	
15	Ibn-Khaldu	Ab-Ar-Rahman					m		Tunis	TS			1332-AD	1406-AD		6	
16	Crescas	Hasdai	ben Abraham				m			ES			1340-AD	1410-AD		6	
17	Nicholas of	Cusa					m		Cusa	GN	Cardinal		1401-AD	1464-AD		5	
18	Ficino	Marsilio					m		Figline	IT			1433-AD	1499-AD		6	
19	Pomponaz	Pietro					m			IT			1462-AD	1525-AD		5	
20	Pico della	Giovanni			Count		m		Mirandol	IT			1463-AD	1494-AD		6	
21	Erasmus	Desiderius					m		Rotterda	HD			1466-AD	1536-AD		6	
22	Machiavell	Niccolo					m		Florence	IT	Statesma	Political	1469-AD	1527-AD		6	
23	Copernicu	Nicolas				Mikolaj	m		Torunac	PL	Mathemat	Optics	1473-AD	1543-AD		6	
24	More	Thomas			Sir/Saint		m		London	UK	Statesma	Scholar	1478-AD	1535-AD		5	

APPENDIX D (Relational appendix)

Convertion from the Semantic Schema to Relational Schema.

HISTORICAL FIGURE
person-id-key:Integer 1:1
date-of-birth: :-2000..2050
date-of-death: :-2000..2050
birth-order:Integer
gender:'m','f'
parent:Boolean
funding:'independent','patron','institution','other'
comments:String
number-of-references:Integer
method:Integer
worked-in—city-id:Integer
born-in—city-id:Integer
worked-in-country--symbol:Char(4)
born-in-country--symbol:Char(4)

PERSON
person-id-key:Integer 1:1
first-name:String
last-name:String
middle-name:String
suffix:String
author-type:String
person-title:String
a-k-a:String

CONTRIBUTION PERIOD
period-id-key:Integer

COUNTRY
name:String 1:1 total
symbol-key:Char(4) 1:1 total
western-since:-2000..2050
western-until:-2000..2050

H-F-PUPIL-OF--PERSON
h-figure--person-id-K:Integer in-key
person-id-K:Integer in-key

CITY
city-id-key:Integer 1:1
name:String
belongs-to-symbol:Char(4)

H-F-HAS-PRACTICED--RELIGION
h-figure--person-id-K:Integer in-key
religion--name-K:String in-key

RELIGION
name-key:String 1:1 total

RELIGION PERIOD
period-id-key:Integer 1:1
of-country--symbol:Char(4)
of-religion--name:String

PERIOD
period-id-key:Integer 1:1
from-year:-2000..2050
to-year:-2000..2050
percentage:1..100

AREA OF CONTRIBUTION
name-key:String 1:1 total
comments:String
is-part-of--name:String

REGION
region-id-key:Integer 1:1
name:String
is-part-of--region-id:Integer

CONTRIBUTION TO AREA
contribution-to-area-id-key:Integer 1:1
major:Boolean total
school-of:String
profession:String
by--person-id:Integer
during--period-id:Integer
of--name:String

COUNTRY-IS-IN--REGION
country--symbol-K:Char(4) in-key
region-id-K:Integer in-key

POPULATION
population-id-key:Integer 1:1
number-in-millions: 0.00...99.99 total
year:-2000..2050 total
c-had-population--symbol:Char(4)

WORLD POPULATION
population-id-key:Integer 1:1

NOBEL PRIZE FIELD

name-key:String 1:1

REFERENCE

Reference-id-key:Integer 1:1
type:'lines','index','other'
number-of-instances:Integer
in—book-id:Integer
to—person-id:Integer

NOBEL PRIZE

nobel-prize-id-key:Integer 1:1
year:1850..2050
in—name:String

BOOK—OTHER-AUTHORS--PERSON

book-id-K:Integer in-key
person-id-K:Integer in-key

N-P-WON-BY—HISTORICAL-FIGURE

nobel-prize-id-K:Integer in-key
h-figure—person-id-K:Integer in-key

BOOK

book-id-keyInteger 1:1
title:String total
date-published:Date
ISBN:String 1:1
first-author—person-id:Integer
published-by—publisher-id:Integer

PUBLISHER

publisher-id-key:Integer 1:1
name:String total
address:String

Script for Creating the Relational database.

```
/* Generated by rddl2sql program version CI-1.2.9207 on Tue May 13 12:25:01 1997
 */
/*modified 12/18/97 by Jenny Rodriguez*/
/*modified 6/29/98 by Tin Ho*/

CREATE SEQUENCE SEQ_rddl2sql;
/************** CATEGORY POPULATION *****************/
CREATE TABLE POPULATION
  ( population_id_key INTEGER
      NOT NULL
   , number_in_millions NUMBER
      NOT NULL
   , year DATE
   , c_had_population__symbol VARCHAR2(3)
  , PRIMARY KEY (population_id_key)  );
COMMENT ON TABLE POPULATION IS 'A record of censuses or estimates of populations of countires
and the world) ';
  CREATE UNIQUE INDEX IND_population1 ON POPULATION(population_id_key);
  COMMENT ON COLUMN POPULATION.population_id_key IS 'The artificial unique identifier for the
objects of POPULATION)';
  COMMENT ON COLUMN POPULATION.number_in_millions IS 'The population)';
  COMMENT ON COLUMN POPULATION.year IS 'The year of the census or estimate of the
population)';
  COMMENT ON COLUMN POPULATION.c_had_population__symbol IS 'Population record of the
country) Full name: country_had_population__symbol.';
/************** CATEGORY REGION *****************/
CREATE TABLE REGION
  ( region_id_key INTEGER
      NOT NULL
   , name VARCHAR(250)
   , is_part_of__region_id INTEGER
  , PRIMARY KEY (region_id_key)  );
COMMENT ON TABLE REGION IS 'A region, sub_region, continent or any other geographical area
which may be of interest) ';
  CREATE UNIQUE INDEX IND_region_id_2 ON REGION(region_id_key);
  COMMENT ON COLUMN REGION.region_id_key IS 'The artificial unique identifier for the objects of
REGION)';
  COMMENT ON COLUMN REGION.name IS 'A name of the region)';
  COMMENT ON COLUMN REGION.is_part_of__region_id IS 'A region may be a part of another
region)';
/************** CATEGORY BOOK *****************/
CREATE TABLE BOOK
  ( book_id_key INTEGER
      NOT NULL
   , title VARCHAR(250)
      NOT NULL
   , date_published DATE
   , ISBN VARCHAR(250)
       UNIQUE
   , first_author__person_id INTEGER
   , published_by__publisher_id INTEGER
```

```
    , person_type VARCHAR(250)
    , book_name VARCHAR2(250)
    , volume INTEGER
    , issue  INTEGER
    , pagenum VARCHAR2(250)
    , frequency VARCHAR2(250)
    , type_of_work VARCHAR2(250)
    , journal_date DATE
  , PRIMARY KEY (book_id_key)   );
COMMENT ON TABLE BOOK IS 'A catalog of books in which references were located) ';
  CREATE UNIQUE INDEX IND_book_id_ke3 ON BOOK(book_id_key);
  COMMENT ON COLUMN BOOK.book_id_key IS 'The artificial unique identifier for the objects of
BOOK)';
  COMMENT ON COLUMN BOOK.title IS 'The title of the book)';
  COMMENT ON COLUMN BOOK.date_published IS 'The date of publication, if known)';
  CREATE UNIQUE INDEX IND_ISBN_____4 ON BOOK(ISBN);
  COMMENT ON COLUMN BOOK.ISBN IS 'The International Standard Book Number, if known)';
  COMMENT ON COLUMN BOOK.first_author__person_id IS 'The main person who wrote the book)';
  COMMENT ON COLUMN BOOK.published_by__publisher_id IS 'The publisher of the book, if
known)';
/************** CATEGORY AREA_OF_CONTRIBUTION ******************/
CREATE TABLE AREA_OF_CONTRIBUTION
  ( name_key VARCHAR(250)
       NOT NULL
    , comments VARCHAR(250)
    , is_part_of__name VARCHAR(250)
  , PRIMARY KEY (name_key)   );
COMMENT ON TABLE AREA_OF_CONTRIBUTION IS 'A catalog of areas of contribution) ';
  CREATE UNIQUE INDEX IND_name_key__5 ON AREA_OF_CONTRIBUTION(name_key);
  COMMENT ON COLUMN AREA_OF_CONTRIBUTION.name_key IS 'The name of the area of
contribution)';
  COMMENT ON COLUMN AREA_OF_CONTRIBUTION.comments IS 'A comment)';
  COMMENT ON COLUMN AREA_OF_CONTRIBUTION.is_part_of__name IS 'The area of which this
is a subarea)';
/************** CATEGORY NOBEL_PRIZE_FIELD ******************/
CREATE TABLE NOBEL_PRIZE_FIELD
  ( name_key VARCHAR(250)
       NOT NULL
  , PRIMARY KEY (name_key)   );
COMMENT ON TABLE NOBEL_PRIZE_FIELD IS 'A catalog of fields in which Nobel Prizes are
awarded) ';
  CREATE UNIQUE INDEX IND_name_key__6 ON NOBEL_PRIZE_FIELD(name_key);
  COMMENT ON COLUMN NOBEL_PRIZE_FIELD.name_key IS 'The name of the field of the Nobel
Prize)';
/************** CATEGORY CITY ******************/
CREATE TABLE CITY
  ( city_id_key INTEGER
       NOT NULL
    , name VARCHAR(250)
    , belongs_to__symbol VARCHAR2(2)
  , PRIMARY KEY (city_id_key)   );
COMMENT ON TABLE CITY IS 'A catalog of cities of the word as of August 1996.) ';
  CREATE UNIQUE INDEX IND_city_id_ke7 ON CITY(city_id_key);
```

```
  COMMENT ON COLUMN CITY.city_id_key IS 'The artificial unique identifier for the objects of
CITY)';
  COMMENT ON COLUMN CITY.name IS 'A modern name of city.)';
  COMMENT ON COLUMN CITY.belongs_to__symbol IS 'A city the city belongs to)';
/************** CATEGORY REFERENCE ******************/
CREATE TABLE REFERENCE
  ( reference_id_key INTEGER
      NOT NULL
   , type CHAR(5)
       CHECK (type IN ( 'lines','index','other' ) )
   , number_of_instances INTEGER
   , to__person_id INTEGER
   , in__book_id INTEGER
  , PRIMARY KEY (reference_id_key)  );
COMMENT ON TABLE REFERENCE IS 'A catalog of references to persons in books) ';
  CREATE UNIQUE INDEX IND_reference_8 ON REFERENCE(reference_id_key);
  COMMENT ON COLUMN REFERENCE.reference_id_key IS 'The artificial unique identifier for the
objects of REFERENCE)';
  COMMENT ON COLUMN REFERENCE.type IS 'The type of reference made. e.g. by counting lines, by
looking in the index)';
  COMMENT ON COLUMN REFERENCE.number_of_instances IS 'The number of references to a
particular historical figure found in the book)';
  COMMENT ON COLUMN REFERENCE.to__person_id IS 'The historical figure who is referenced)';
  COMMENT ON COLUMN REFERENCE.in__book_id IS 'The book in which the reference was
found.)';
/************** CATEGORY PUBLISHER ******************/
CREATE TABLE PUBLISHER
  ( publisher_id_key INTEGER
      NOT NULL
   , name VARCHAR(250)
      NOT NULL
   , address VARCHAR(250)
  , PRIMARY KEY (publisher_id_key)  );
COMMENT ON TABLE PUBLISHER IS 'A catalog of publishers) ';
  CREATE UNIQUE INDEX IND_publisher_9 ON PUBLISHER(publisher_id_key);
  COMMENT ON COLUMN PUBLISHER.publisher_id_key IS 'The artificial unique identifier for the
objects of PUBLISHER)';
  COMMENT ON COLUMN PUBLISHER.name IS 'The name of the publisher)';
  COMMENT ON COLUMN PUBLISHER.address IS 'The address of the publisher)';
/************** CATEGORY PERSON ******************/
CREATE TABLE PERSON
  ( person_id_key INTEGER
      NOT NULL
   , first_name VARCHAR(250)
   , last_name VARCHAR(250)
   , middle_name VARCHAR(250)
   , suffix VARCHAR(250)
   , author_type VARCHAR2(250)
   , person_title VARCHAR2(250)
   , a_k_a VARCHAR2(250)
    /**# , percentage INTEGER  #**/
  , PRIMARY KEY (person_id_key)  );
COMMENT ON TABLE PERSON IS 'A record of all persons relevant to this database) ';
```

```
  CREATE UNIQUE INDEX IND_person_id_10 ON PERSON(person_id_key);
  COMMENT ON COLUMN PERSON.person_id_key IS 'The artificial unique identifier for the objects of
PERSON)';
  COMMENT ON COLUMN PERSON.first_name IS 'The first name of a person)';
  COMMENT ON COLUMN PERSON.last_name IS 'The last name of a person)';
  COMMENT ON COLUMN PERSON.middle_name IS 'The middle name of a person. Convention: name
descriptors such as Queen of England are placed in the attribute middle_name.)';
  COMMENT ON COLUMN PERSON.suffix IS 'The suffix of a name, e.g. Jr in Martin Luther King Jr., II
in Elizabeth II.)';
/************** CATEGORY COUNTRY ******************/
CREATE TABLE COUNTRY
  ( symbol_key VARCHAR2(2)
      NOT NULL
   , name VARCHAR(250)
   , western_since Number(*,0) /**# CHECK (western_since BETWEEN -2000 AND 2050) #**/
   , western_until Number(*,0) /**# CHECK (western_until BETWEEN -2000 AND 2050) #**/
  , PRIMARY KEY (symbol_key)  );
COMMENT ON TABLE COUNTRY IS 'A catalog of countries of the world as of August 1996. Each
country is considered in its August 1996 borders, even though maps may have changed.) ';
  CREATE UNIQUE INDEX IND_symbol_key11 ON COUNTRY(symbol_key);
  COMMENT ON COLUMN COUNTRY.symbol_key IS 'The international 2_letter symbol of the
country, e.g. US, RU, AU, CA)';
  COMMENT ON COLUMN COUNTRY.name IS 'The modern name of the country)';
  COMMENT ON COLUMN COUNTRY.western_since IS 'The beginning year in which the country was
considered part of Western Civilization)';
  COMMENT ON COLUMN COUNTRY.western_until IS 'The ending year in which the country was
considered part of Western Civilization)';
/************** CATEGORY RELIGION ******************/
CREATE TABLE RELIGION
  ( name_key VARCHAR(250)
      NOT NULL
  , PRIMARY KEY (name_key)  );
COMMENT ON TABLE RELIGION IS 'A catalog of religious denominations widespread in some
countries) ';
  CREATE UNIQUE INDEX IND_name_key__12 ON RELIGION(name_key);
  COMMENT ON COLUMN RELIGION.name_key IS 'The name of a religion)';
/************** CATEGORY PERIOD ******************/
CREATE TABLE PERIOD
  ( period_id_key INTEGER
      NOT NULL
   , from_year Number(*,0) /**# CHECK (from_year BETWEEN -2000 AND 2050)#**/
   , to_year Number(*,0) /**# CHECK (to_year BETWEEN -2000 AND 2050) #**/
   , percentage Number(*,0) CHECK (percentage BETWEEN 1 AND 100)
  , PRIMARY KEY (period_id_key)  );
COMMENT ON TABLE PERIOD IS 'A period of time) ';
  CREATE UNIQUE INDEX IND_period_id_13 ON PERIOD(period_id_key);
  COMMENT ON COLUMN PERIOD.period_id_key IS 'The artificial unique identifier for the objects of
PERIOD)';
  COMMENT ON COLUMN PERIOD.from_year IS 'The year of period beginning.)';
  COMMENT ON COLUMN PERIOD.to_year IS 'The year of period end.)';
  COMMENT ON COLUMN PERIOD.percentage IS 'An estimate of percentage of the population of the
country which practiced the religion or contribution of given person in given period)';
/************** CATEGORY NOBEL_PRIZE ******************/
```

```
CREATE TABLE NOBEL_PRIZE
  ( nobel_prize_id_key INTEGER
      NOT NULL
    , year DATE
    , in__name VARCHAR(250)
  , PRIMARY KEY (nobel_prize_id_key)   );
COMMENT ON TABLE NOBEL_PRIZE IS 'A catalog of Nobel Prizes) ';
  CREATE UNIQUE INDEX IND_nobel_priz14 ON NOBEL_PRIZE(nobel_prize_id_key);
  COMMENT ON COLUMN NOBEL_PRIZE.nobel_prize_id_key IS 'The artificial unique identifier for
the objects of NOBEL_PRIZE)';
  COMMENT ON COLUMN NOBEL_PRIZE.year IS 'The year in which a Nobel Prize was awarded to a
particular person)';
  COMMENT ON COLUMN NOBEL_PRIZE.in__name IS 'The field in which the Nobel Prize was won)';
/************** CATEGORY CONTRIBUTION_TO_AREA ******************/
CREATE TABLE CONTRIBUTION_TO_AREA
  ( contribution_to_area_id_key INTEGER
      NOT NULL
    , major CHAR(1) CHECK (major IN ('y','n'))
      NOT NULL
    , of__name VARCHAR(250)
    , by__person_id INTEGER
    , during__period_id INTEGER
    , school_of VARCHAR2(250)
    , profession VARCHAR2(250)
  , PRIMARY KEY (contribution_to_area_id_key)   );
COMMENT ON TABLE CONTRIBUTION_TO_AREA IS 'A record of associations of historical figures
with specific areas) ';
  CREATE UNIQUE INDEX IND_contributi15 ON
CONTRIBUTION_TO_AREA(contribution_to_area_id_key);
  COMMENT ON COLUMN CONTRIBUTION_TO_AREA.contribution_to_area_id_key IS 'The
artificial unique identifier for the objects of CONTRIBUTION_TO_AREA)';
  COMMENT ON COLUMN CONTRIBUTION_TO_AREA.major IS 'This attribute distinguishes a
contribution considered a major contribution)';
  COMMENT ON COLUMN CONTRIBUTION_TO_AREA.of__name IS 'The area to which a
contribution was made)';
  COMMENT ON COLUMN CONTRIBUTION_TO_AREA.by__person_id IS 'The historical figure by
whom the contribution was made)';
  COMMENT ON COLUMN CONTRIBUTION_TO_AREA.during__period_id IS 'The period during
which a contribution was made)';
/************** CATEGORY HISTORICAL_FIGURE ******************/
CREATE TABLE HISTORICAL_FIGURE
  ( person_id_key INTEGER
      NOT NULL
    , date_of_birth DATE
    , date_of_death DATE
    , birth_order NUMBER
    , parent CHAR(1) CHECK (parent IN ('y','n'))
    , funding CHAR(11)
        CHECK (funding IN ( 'independent','patron','institution','other' ) )
    , comments VARCHAR(250)
    , worked_in__city_id NUMBER
    , born_in__city_id NUMBER
    , pupil_of VARCHAR2(250)
```

```
    , country_worked_in VARCHAR2(250)
    , country_born_in VARCHAR2(250)
    , reference_number NUMBER
  , PRIMARY KEY (person_id_key)   );
COMMENT ON TABLE HISTORICAL_FIGURE IS 'Historical figures are persons who are subjects of
this study.)';
  CREATE UNIQUE INDEX IND_person_id_16 ON HISTORICAL_FIGURE(person_id_key);
  COMMENT ON COLUMN HISTORICAL_FIGURE.person_id_key IS 'The artificial unique identifier
for the objects of PERSON)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.date_of_birth IS 'The date of birth, if known)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.date_of_death IS 'The date of death, if known)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.birth_order IS 'A generation which the historical
figure belongs to)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.parent IS 'True if a parent of the historical figure
was contributing to the same field)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.funding IS 'The type of funding the person
received)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.comments IS 'Comment for Historical Figure. May
be used also to indicate that the country of birth is different from country of work)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.worked_in__city_id IS 'The modern city within
which modern borders the person was working)';
  COMMENT ON COLUMN HISTORICAL_FIGURE.born_in__city_id IS 'The modern city within which
modern borders the person was born)';
/************** CATEGORY CONTRIBUTION_PERIOD *****************/
CREATE TABLE CONTRIBUTION_PERIOD
  ( period_id_key INTEGER
      NOT NULL
  , PRIMARY KEY (period_id_key)   );
COMMENT ON TABLE CONTRIBUTION_PERIOD IS 'A record of contribution of a particular
historical person to specific area during time periods.)';
  CREATE UNIQUE INDEX IND_period_id_17 ON CONTRIBUTION_PERIOD(period_id_key);
  COMMENT ON COLUMN CONTRIBUTION_PERIOD.period_id_key IS 'The artificial unique
identifier for the objects of PERIOD)';
/************** CATEGORY RELIGION_PERIOD ******************/
CREATE TABLE RELIGION_PERIOD
  ( period_id_key INTEGER
      NOT NULL
    , of_religion__name VARCHAR(250)
    , of_country__symbol VARCHAR2(2)
  , PRIMARY KEY (period_id_key)   );
COMMENT ON TABLE RELIGION_PERIOD IS 'A record of religions of particular countries during
specific time periods. A country can have several widespread religions at the same time. In addition,
countries change their religion over time.)';
  CREATE UNIQUE INDEX IND_period_id_18 ON RELIGION_PERIOD(period_id_key);
  COMMENT ON COLUMN RELIGION_PERIOD.period_id_key IS 'The artificial unique identifier for
the objects of PERIOD)';
  COMMENT ON COLUMN RELIGION_PERIOD.of_religion__name IS 'The subject religion during a
particular period)';
  COMMENT ON COLUMN RELIGION_PERIOD.of_country__symbol IS 'The subject country)';
/************** CATEGORY WORLD_POPULATION *****************/
/**## tin 1998 jun 19 ##**/
/**## original table name was world_population, now changed to world_population2 ##**/
/**## the original key had name population_id_key, now changed to WORLD_population_id_key ##**/
```

```
CREATE TABLE WORLD_POPULATION2
  ( world_population_id_key INTEGER
      NOT NULL
  , number_in_millions INTEGER
  , year DATE
  , PRIMARY KEY (WORLD_population_id_key)  );
COMMENT ON TABLE WORLD_POPULATION2 IS 'Records of world population estimates on various
dates)';
  CREATE UNIQUE INDEX IND_population19 ON
WORLD_POPULATION2(WORLD_population_id_key);
  COMMENT ON COLUMN WORLD_POPULATION2.WORLD_population_id_key IS 'The artificial
unique identifier for the objects of POPULATION)';
/************** CATEGORY H_F_HAS_PRACTICED__RELIGION ******************/
CREATE TABLE H_F_HAS_PRACTICED__RELIGION
  ( h_figure__person_id_K INTEGER
      NOT NULL
   , religion__name_K VARCHAR(250)
      NOT NULL
  , PRIMARY KEY (h_figure__person_id_K, religion__name_K)  );
CREATE UNIQUE INDEX IND_H_F_HAS_PR20 ON
H_F_HAS_PRACTICED__RELIGION(h_figure__person_id_K,religion__name_K);
COMMENT ON TABLE H_F_HAS_PRACTICED__RELIGION IS 'represents m:m relation
has_practiced (The religion practiced by a historical figure. A person may have practiced several religions)
between HISTORICAL_FIGURE (Historical figures are persons who are subjects of this study.) and
RELIGION (A catalog of reli';
  COMMENT ON COLUMN H_F_HAS_PRACTICED__RELIGION.h_figure__person_id_K IS '" Full
name: historical_figure__person_id_K.';
  COMMENT ON COLUMN H_F_HAS_PRACTICED__RELIGION.religion__name_K IS '"';
/************** CATEGORY N_P_WON_BY__HISTORICAL_FIGURE ******************/
CREATE TABLE N_P_WON_BY__HISTORICAL_FIGURE
  ( nobel_prize_id_K INTEGER
      NOT NULL
   , h_figure__person_id_K INTEGER
      NOT NULL
  , PRIMARY KEY (nobel_prize_id_K, h_figure__person_id_K)  );
CREATE UNIQUE INDEX IND_N_P_WON_BY21 ON
N_P_WON_BY__HISTORICAL_FIGURE(nobel_prize_id_K,h_figure__person_id_K);
COMMENT ON TABLE N_P_WON_BY__HISTORICAL_FIGURE IS 'represents m:m relation won_by
(The historical figure who won the Nobel Prize) between NOBEL_PRIZE (A catalog of Nobel Prizes)  and
HISTORICAL_FIGURE (Historical figures are persons who are subjects of this study.) ) Full name:
NOBEL_PRIZE__WON_BY__HISTOR';
  COMMENT ON COLUMN N_P_WON_BY__HISTORICAL_FIGURE.nobel_prize_id_K IS '"';
  COMMENT ON COLUMN N_P_WON_BY__HISTORICAL_FIGURE.h_figure__person_id_K IS '"
Full name: historical_figure__person_id_K.';
/************** CATEGORY BOOK__OTHER_AUTHORS__PERSON ******************/
CREATE TABLE BOOK__OTHER_AUTHORS__PERSON
  ( book_id_K INTEGER
      NOT NULL
   , person_id_K INTEGER
      NOT NULL
  , PRIMARY KEY (book_id_K, person_id_K)  );
CREATE UNIQUE INDEX IND_BOOK__OTHE22 ON
BOOK__OTHER_AUTHORS__PERSON(book_id_K,person_id_K);
```

```
COMMENT ON TABLE BOOK__OTHER_AUTHORS__PERSON IS 'represents m:m relation
other_authors (The co_authors of the book, other than the first author.) between BOOK (A catalog of books
in which references were located)  and PERSON (A record of all persons relevant to this database)  )';
  COMMENT ON COLUMN BOOK__OTHER_AUTHORS__PERSON.book_id_K IS "';
  COMMENT ON COLUMN BOOK__OTHER_AUTHORS__PERSON.person_id_K IS "';
/************** CATEGORY COUNTRY__IS_IN__REGION *****************/
CREATE TABLE COUNTRY__IS_IN__REGION
  ( country__symbol_K VARCHAR2(2)
      NOT NULL
   , region_id_K INTEGER
      NOT NULL
  , PRIMARY KEY (country__symbol_K, region_id_K)   );
CREATE UNIQUE INDEX IND_COUNTRY__I23 ON
COUNTRY__IS_IN__REGION(country__symbol_K,region_id_K);
COMMENT ON TABLE COUNTRY__IS_IN__REGION IS 'represents m:m relation is_in (A region the
country is in) between COUNTRY (A catalog of countries of the world as of August 1996. Each country is
considered in its August 1996 borders, even though maps may have changed.)  and REGION (A region,
sub_region';
  COMMENT ON COLUMN COUNTRY__IS_IN__REGION.country__symbol_K IS "';
  COMMENT ON COLUMN COUNTRY__IS_IN__REGION.region_id_K IS "';
/************** REFERENCE CONSTRAINTS *****************/
ALTER TABLE POPULATION ADD (
  FOREIGN KEY (c_had_population__symbol)
    REFERENCES COUNTRY (symbol_key)
  );
ALTER TABLE REGION ADD (
  FOREIGN KEY (is_part_of__region_id)
    REFERENCES REGION (region_id_key)
  );
ALTER TABLE BOOK ADD (
  FOREIGN KEY (first_author__person_id)
    REFERENCES PERSON (person_id_key)
 , FOREIGN KEY (published_by__publisher_id)
    REFERENCES PUBLISHER (publisher_id_key)
  );
ALTER TABLE AREA_OF_CONTRIBUTION ADD (
  FOREIGN KEY (is_part_of__name)
    REFERENCES AREA_OF_CONTRIBUTION (name_key)
  );
ALTER TABLE CITY ADD (
  FOREIGN KEY (belongs_to__symbol)
    REFERENCES COUNTRY (symbol_key)
  );
ALTER TABLE REFERENCE ADD (
  FOREIGN KEY (to__person_id)
    REFERENCES HISTORICAL_FIGURE (person_id_key)
 , FOREIGN KEY (in__book_id)
    REFERENCES BOOK (book_id_key)
  );
ALTER TABLE NOBEL_PRIZE ADD (
  FOREIGN KEY (in__name)
    REFERENCES NOBEL_PRIZE_FIELD (name_key)
  );
```

```
ALTER TABLE CONTRIBUTION_TO_AREA ADD (
  FOREIGN KEY (of__name)
    REFERENCES AREA_OF_CONTRIBUTION (name_key)
 , FOREIGN KEY (by__person_id)
    REFERENCES HISTORICAL_FIGURE (person_id_key)
 , FOREIGN KEY (during__period_id)
    REFERENCES CONTRIBUTION_PERIOD (period_id_key)
  );
ALTER TABLE HISTORICAL_FIGURE ADD (
  FOREIGN KEY (person_id_key)
    REFERENCES PERSON (person_id_key)
  );
ALTER TABLE CONTRIBUTION_PERIOD ADD (
  FOREIGN KEY (period_id_key)
    REFERENCES PERIOD (period_id_key)
  );
ALTER TABLE RELIGION_PERIOD ADD (
  FOREIGN KEY (period_id_key)
    REFERENCES PERIOD (period_id_key)
 , FOREIGN KEY (of_religion__name)
    REFERENCES RELIGION (name_key)
 , FOREIGN KEY (of_country__symbol)
    REFERENCES COUNTRY (symbol_key)
  );

/**## table name world_population and key name population_id_key changed ##**/
ALTER TABLE WORLD_POPULATION2 ADD (
  FOREIGN KEY (WORLD_population_id_key)
    REFERENCES POPULATION (population_id_key)
  );

ALTER TABLE H_F_HAS_PRACTICED__RELIGION ADD (
  FOREIGN KEY (h_figure__person_id_K)
    REFERENCES HISTORICAL_FIGURE (person_id_key)
 , FOREIGN KEY (religion__name_K)
    REFERENCES RELIGION (name_key)
  );
ALTER TABLE N_P_WON_BY__HISTORICAL_FIGURE ADD (
  FOREIGN KEY (nobel_prize_id_K)
    REFERENCES NOBEL_PRIZE (nobel_prize_id_key)
 , FOREIGN KEY (h_figure__person_id_K)
    REFERENCES HISTORICAL_FIGURE (person_id_key)
  );
ALTER TABLE BOOK__OTHER_AUTHORS__PERSON ADD (
  FOREIGN KEY (book_id_K)
    REFERENCES BOOK (book_id_key)
 , FOREIGN KEY (person_id_K)
    REFERENCES PERSON (person_id_key)
  );
ALTER TABLE COUNTRY__IS_IN__REGION ADD (
  FOREIGN KEY (country__symbol_K)
    REFERENCES COUNTRY (symbol_key)
 , FOREIGN KEY (region_id_K)
```

```
  REFERENCES REGION (region_id_key)
 );


/**#   changes by tin  #**/

-- table alteration script by tin.  1998 march 9
-- I am not inventing these changes to the stucture of the tables.
-- jenny did the changes before, i am just creating a log/script
-- so that other user can use this sql script to create a copy
-- of the president DB under their account.

-- these changes done after CHECK constrain removed
ALTER TABLE country MODIFY western_since DATE;
ALTER TABLE country MODIFY western_until DATE;
ALTER TABLE period  MODIFY from_year DATE;
ALTER TABLE period  MODIFY to_year DATE;

-- these are just some attribute change (in number of chars)
ALTER TABLE country MODIFY symbol_key VARCHAR2(3);
ALTER TABLE country__is_in__region MODIFY country__symbol_k VARCHAR2(3);
ALTER TABLE religion_period MODIFY of_country__symbol VARCHAR2(3);

ALTER TABLE n_p_won_by__historical_figure ADD ( dummy VARCHAR2(250) );

-- last minute change before jenny left.
ALTER TABLE historical_figure ADD ( method NUMBER );
```

Scripts for Populating the Relational database.

Script for loading the Person table (person.ldr)

```
-- fieldname    SEQUENCE( MAX, 1), to create sequence number, +1 each time
-- decided to use pre created sequence number instead, as each line will
-- be accessed to populate different table

-- don't think can use comma delimited, as some middle field are not to be populated yet.
-- so use .prn format, think we only have one file to work with only anyway

-- last updated by tin, 1997 nov 22

-- find out from maxim/marina whether suffix, empty string any problem?
-- change the into table <table_name> field for final version

LOAD DATA
APPEND
INTO TABLE person
(
 person_id_key    POSITION(  1- 4 ),
 first_name       POSITION(  6-24 ),
 last_name        POSITION( 25-43 ),
 middle_name      POSITION( 44-72 ),
 suffix           POSITION( 73-90 )
)
```

Script for loading the Country table (country.ldr)

```
-- this ldr file read the country coded from the file countries.numbered
-- (originally in countries.xls) into the table country in the
-- president database in oracle

-- last modified 1997 dec 13  by tin

LOAD DATA
APPEND
INTO TABLE country
(
 symbol_key       POSITION( 28-31 ),
 name             POSITION(  6-27 )
)
```

Script for loading the City table (city.ldr)

```
-- quick and dirty for now,
-- fix later

-- this file is intented to populate the table city
-- data are extracted from the pres.xls and scientist.xls files
-- the city/country table is then saved as a prn file,
-- unix sort -u is run on the resulting prn file to extract only the
-- unique records, then tinscript is run on the resulting file to
-- add city id to them.

-- important, populate city and country before moving on to the
-- population of other tables that refer to this,
-- it will be hard to reload the table w/o deleting the dependend records
-- eg in the historical_fiugres table.

-- on 1998 jan 26, data from scientist.xls was extracted just to test the
-- script, note that some 60 records have been rejected cuz the country
-- table did not have the parent key (ie missing country code)
-- lastly, make sure the data has no error, like London UK and London US.


LOAD DATA
APPEND
INTO TABLE city
(
 city_id_key        POSITION( 1- 4 ),
 name               POSITION( 6-26 ),
 belongs_to__symbol  POSITION( 27-28)
)
```

Script for loading the Area-of-contribution table (area_of_contribution.ldr)

```
-- this file is intented to populate the area_to_contribution table
-- data are extracted from the contribution field from the .xls file
-- that contain data mostly for the person table.

-- sample source datafile is now:
--  ../formatted-data/contribution_to_area.ldr.sample_data


-- data files include:
--  ../formatted-data/area_of_contribution.numbered

LOAD DATA
APPEND
INTO TABLE area_of_contribution
(
 name_key        POSITION(  6-36 )
)
```

Script for loading the Contribution-to-area table (contribution_to_area.ldr)

```
-- last modified 1997 dec 13 by tin

-- this file is intented to populate the contribution_to_area table
-- data are extracted from the contribution field from the .xls file
-- that contain data mostly for the person table.

-- sample source datafile is now:
--  ../formatted-data/short-pres-data.numbered


-- data files include:
--  ../formatted-data/pres.numbered


-- for some odd reason, can't populate this table
-- from the contribution_to_area.test.sql file,
-- the problem seems to be some constrain violation (fk?)

-- should be fixed, problem being major = 'y' or 'n' only

LOAD DATA
-- REPLACE
APPEND
INTO TABLE contribution_to_area
(
 contribution_to_area_id_key   SEQUENCE( MAX, 1),
 major                         CONSTANT 'y',
 of__name                      POSITION( 123-140 ),
 by__person_id                 POSITION( 1- 4 )
)
```

APPENDIX E (Semantic C++ appendix)

Program to create the Western Culture Database (presicreate.cpp)

```
/*
 * Created by Raquel Ghersgorin
 * Last Modification: June 23rd. 1998
 */
#include <stdlib.h>
#include <stdio.h>
#include "sdb3.h"

void SchEd_CreateDataBase ( const char * fileName, TDataBase * theDatabase )
{
 theDatabase -> Transaction_Begin ( );

// Creating New types
   NewNumberClass ( theDatabase, "0.00...99.99",0.00 ,99.99 ,0.01 ,"") ;
   NewIntegerClass ( theDatabase, "1850..2050", 1850, 2051, "range" ) ;
   NewIntegerClass ( theDatabase, "-2000..2050", -2000, 2050, "range" ) ;
   NewIntegerClass ( theDatabase, "1..100", 1, 100, "" ) ;
   NewStringClass ( theDatabase, "Char(4)", 4, "", "" ) ;

// Creating categories and their attributes
   NewCategory ( theDatabase, "PERSON", "A record of all persons relevant to this database" ) ;
   NewRelation ( theDatabase, "FIRST_NAME", "PERSON", "String", CC_MANY_TO_ONE,
"The first name of a person") ;
   NewRelation ( theDatabase, "LAST_NAME", "PERSON", "String", CC_MANY_TO_ONE,
"The last name of a person") ;
   NewRelation ( theDatabase, "MIDDLE_NAME", "PERSON", "String", CC_MANY_TO_ONE,
"The middle name of a person. Convention: name descriptors such as Queen of England are placed
in the attribute middle-name.") ;
   NewRelation ( theDatabase, "SUFFIX", "PERSON", "String", CC_MANY_TO_ONE, "The
suffix of a name, e.g. Jr in Martin Luther King Jr., II in Elizabeth II.") ;
   NewRelation ( theDatabase, "AUTHOR_TYPE", "PERSON", "String", CC_MANY_TO_ONE,
"The type of the person when is an author of a book") ;
   NewRelation ( theDatabase, "PERSON_TITLE", "PERSON", "String", CC_MANY_TO_ONE,
"The title of a person") ;
   NewRelation ( theDatabase, "A_K_A", "PERSON", "String", CC_MANY_TO_ONE, "The
name as the person is known as  ") ;

   NewCategory ( theDatabase, "CONTRIBUTION_TO_AREA", "A record of associations of
historical figures with specific areas" ) ;
   NewRelation ( theDatabase, "MAJOR", "CONTRIBUTION_TO_AREA", "String",
CC_MANY_TO_ONE + CC_TOTAL, "This attribute distinguishes a contribution considered a
major contribution") ;
  // NewRelation ( theDatabase, "MAJOR", "CONTRIBUTION_TO_AREA", "Boolean",
CC_MANY_TO_ONE + CC_TOTAL, "This attribute distinguishes a contribution considered a
major contribution") ;
    NewRelation ( theDatabase, "SCHOOL_OF", "CONTRIBUTION_TO_AREA", "String",
CC_MANY_TO_ONE, "The school to where the Historical Figure made the contribution") ;
   NewRelation ( theDatabase, "PROFESSION", "CONTRIBUTION_TO_AREA", "String",
CC_MANY_TO_ONE, "The profession to which the Historical Figure made the contribution") ;
```

```
NewCategory ( theDatabase, "AREA_OF_CONTRIBUTION", "A catalog of areas of contribution" ) ;
NewRelation ( theDatabase, "NAME", "AREA_OF_CONTRIBUTION", "String", CC_ONE_TO_ONE + CC_TOTAL, "The name of the area of contribution") ;
NewRelation ( theDatabase, "COMMENTS", "AREA_OF_CONTRIBUTION", "String", CC_MANY_TO_ONE, "A comment") ;

NewCategory ( theDatabase, "RELIGION", "A catalog of religious denominations widespread in some countries" ) ;
NewRelation ( theDatabase, "NAME", "RELIGION", "String", CC_ONE_TO_ONE + CC_TOTAL, "The name of a religion") ;

NewCategory ( theDatabase, "RELIGION_PERIOD", "A record of religions of particular countries during specific time periods. A country can have several widespread religions at the same time. In addition, countries change their religion over time." ) ;

NewCategory ( theDatabase, "PERIOD", "A period of time" ) ;
NewRelation ( theDatabase, "FROM_YEAR", "PERIOD", "-2000..2050", CC_MANY_TO_ONE, "The year of period beginning.") ;
NewRelation ( theDatabase, "TO_YEAR", "PERIOD", "-2000..2050", CC_MANY_TO_ONE, "The year of period end.") ;
NewRelation ( theDatabase, "PERCENTAGE", "PERIOD", "1..100", CC_MANY_TO_ONE, "An estimate of percentage of the population of the country which practiced the religion or percentage of people who contributed on a specific area during a particular period") ;

NewCategory ( theDatabase, "CONTRIBUTION_PERIOD", "A record of contribution of a particular historical person to specific area during time periods." ) ;

NewCategory ( theDatabase, "HISTORICAL_FIGURE", "Historical figures are persons who are subjects of this study." ) ;
NewRelation ( theDatabase, "DATE_OF_BIRTH", "HISTORICAL_FIGURE", "-2000..2050", CC_MANY_TO_ONE, "The date of birth, if known") ;
NewRelation ( theDatabase, "DATE_OF_DEATH", "HISTORICAL_FIGURE", "-2000..2050", CC_MANY_TO_ONE, "The date of death, if known") ;
NewRelation ( theDatabase, "BIRTH_ORDER", "HISTORICAL_FIGURE", "Integer", CC_MANY_TO_ONE, ") (A generation which the historical figure belongs to, they are ordinals such as 1,2,3,4,5.., they represent whether the person is a first born, second born") ;

//NewRelation ( theDatabase, "PARENT", "HISTORICAL_FIGURE", "Boolean", CC_MANY_TO_ONE, "") ;
NewRelation ( theDatabase, "GENDER", "HISTORICAL_FIGURE", "String", CC_MANY_TO_ONE, "The gender of the person , i.e. m:male, f:female") ;
NewRelation ( theDatabase, "PARENT", "HISTORICAL_FIGURE", "String", CC_MANY_TO_ONE, "The parent's occupation. If it is true it means that the person's father did the same work as the person ") ;
NewRelation ( theDatabase, "FUNDING", "HISTORICAL_FIGURE", "String", CC_MANY_TO_ONE, "The type of funding the person received") ;
NewRelation ( theDatabase, "COMMENTS", "HISTORICAL_FIGURE", "String", CC_MANY_TO_ONE, "Comment for Historical Figure. May be used also to indicate that the country of birth is different from country of work") ;
NewRelation ( theDatabase, "REFERENCE_NUMBER", "HISTORICAL_FIGURE", "Integer", CC_MANY_TO_ONE, "The number of references of that Historical Figure in the literature") ;
```

NewRelation (theDatabase, "METHOD", "HISTORICAL_FIGURE", "Integer", CC_MANY_TO_ONE, "The method's number used to collect the information about an Historical Figure") ;

NewCategory (theDatabase, "REGION", "A region, sub-region, continent, or any other geographical area which may be of interest") ;
NewRelation (theDatabase, "NAME", "REGION", "String", CC_MANY_TO_ONE, "A name of the Region") ;

NewCategory (theDatabase, "WORLD_POPULATION", "") ;

NewCategory (theDatabase, "POPULATION", "A record of censuses or estimates of population of countries and the world") ;
NewRelation (theDatabase, "NUMBER_IN_MILLIONS", "POPULATION", "0.00...99.99", CC_MANY_TO_ONE + CC_TOTAL, "The population") ;
NewRelation (theDatabase, "YEAR", "POPULATION", "-2000..2050", CC_MANY_TO_ONE + CC_TOTAL, "The year of census or estimate of the population") ;

NewCategory (theDatabase, "CITY", "The record of all cities relevant to this study") ;
NewRelation (theDatabase, "NAME", "CITY", "String", CC_MANY_TO_ONE, "") ;

NewCategory (theDatabase, "COUNTRY", "Table of countries relevant to this study") ;
NewRelation (theDatabase, "NAME", "COUNTRY", "String", CC_ONE_TO_ONE + CC_TOTAL, "The modern name of the Country") ;
NewRelation (theDatabase, "SYMBOL", "COUNTRY", "Char(4)", CC_ONE_TO_ONE + CC_TOTAL, "The international 4-letter symbol of the country, e.g. US,RU,AU,CA,USVI") ;
NewRelation (theDatabase, "WESTERN_SINCE", "COUNTRY", "-2000..2050", CC_MANY_TO_ONE, "The begining year in which the country was considered part of Western Civilization") ;
NewRelation (theDatabase, "WESTERN_UNTIL", "COUNTRY", "-2000..2050", CC_MANY_TO_ONE, "The begining year in which the country was considered part of Western Civilization") ;

NewCategory (theDatabase, "NOBEL_PRIZE", "A catalog of Nobel Prizes") ;
NewRelation (theDatabase, "YEAR", "NOBEL_PRIZE", "1850..2050", CC_MANY_TO_ONE, "The year in which a Nobel Prize was awarded to a particular person") ;

NewCategory (theDatabase, "NOBEL_PRIZE_FIELD", "A catalog of fields in which Nobel Prizes are awarded") ;
NewRelation (theDatabase, "NAME", "NOBEL_PRIZE_FIELD", "String", CC_ONE_TO_ONE + CC_TOTAL, "The name of the field of the Nobel Prize") ;

NewCategory (theDatabase, "REFERENCE", "A catalog of references to persons in books") ;
NewRelation (theDatabase, "TYPE", "REFERENCE", "String", CC_MANY_TO_ONE, "The type of reference made. e.g. by counting lines, by looking in the index") ;
NewRelation (theDatabase, "NUMBER_OF_INSTANCES", "REFERENCE", "Integer", CC_MANY_TO_ONE, "The number of references to a particular historical figure found in the book") ;

NewCategory (theDatabase, "BOOK", "A catalog of books in which references were located") ;
NewRelation (theDatabase, "TITLE", "BOOK", "String", CC_MANY_TO_ONE + CC_TOTAL, "The title of the book") ;

```
  NewRelation ( theDatabase, "DATE_PUBLISHED", "BOOK", "Date", CC_MANY_TO_ONE,
"The date of publication, if known") ;
  NewRelation ( theDatabase, "ISBN", "BOOK", "String", CC_ONE_TO_ONE, "The
International Standard Book Number, if known") ;
 // NewRelation ( theDatabase, "TITLE_ARTICLE", "BOOK", "String", CC_MANY_TO_ONE,
"The title of the article where the reference was found") ;
 // NewRelation ( theDatabase, "VOLUME", "BOOK", "String", CC_MANY_TO_ONE, "The
volume of the journal where the reference was found") ;
 // NewRelation ( theDatabase, "ISSUE", "BOOK", "String", CC_MANY_TO_ONE, "The issue
of the journal where the reference was found") ;
 // NewRelation ( theDatabase, "PAGENUM", "BOOK", "String", CC_MANY_TO_ONE, "The
number of pages referencing the Historical Figure") ;
 // NewRelation ( theDatabase, "FREQUENCY", "BOOK", "String", CC_MANY_TO_ONE, "") ;
 // NewRelation ( theDatabase, "TYPE_OF_WORK", "BOOK", "String", CC_MANY_TO_ONE,
"") ;
 // NewRelation ( theDatabase, "JOURNAL_DATE", "BOOK", "Date", CC_MANY_TO_ONE,
"") ;

  NewCategory ( theDatabase, "PUBLISHER", "A catalog of publishers" ) ;
  NewRelation ( theDatabase, "NAME", "PUBLISHER", "String", CC_MANY_TO_ONE +
CC_TOTAL, "The name of the publisher") ;
  NewRelation ( theDatabase, "ADDRESS", "PUBLISHER", "String", CC_MANY_TO_ONE,
"The address of the publisher") ;


// Creating inheritance
  MakeInheritance ( "RELIGION_PERIOD", "PERIOD", theDatabase ) ;
  MakeInheritance ( "CONTRIBUTION_PERIOD", "PERIOD", theDatabase ) ;
  MakeInheritance ( "HISTORICAL_FIGURE", "PERSON", theDatabase ) ;
  MakeInheritance ( "WORLD_POPULATION", "POPULATION", theDatabase ) ;

// Creating relations
  NewRelation ( theDatabase, "OF", "CONTRIBUTION_TO_AREA",
"AREA_OF_CONTRIBUTION", CC_MANY_TO_ONE, "The area to which a contribution was
made") ;
  NewRelation ( theDatabase, "BY", "CONTRIBUTION_TO_AREA",
"HISTORICAL_FIGURE", CC_MANY_TO_ONE, "The historical figure by whom the
contribution was made") ;
  NewRelation ( theDatabase, "DURING_PERIOD", "CONTRIBUTION_TO_AREA",
"CONTRIBUTION_PERIOD", CC_MANY_TO_ONE, "The period during which a contribution
was made") ;
  NewRelation ( theDatabase, "IS_PART_OF", "AREA_OF_CONTRIBUTION",
"AREA_OF_CONTRIBUTION", CC_MANY_TO_ONE, "The area of which this is a subarea") ;
  NewRelation ( theDatabase, "OF_RELIGION", "RELIGION_PERIOD", "RELIGION",
CC_MANY_TO_ONE, "The subject religion during a particular period") ;
  NewRelation ( theDatabase, "OF_COUNTRY", "RELIGION_PERIOD", "COUNTRY",
CC_MANY_TO_ONE, "The subject country") ;
  NewRelation ( theDatabase, "PUPIL_OF", "HISTORICAL_FIGURE", "PERSON",
CC_MANY_TO_MANY, "The theacher of that Historical Figure") ;
  NewRelation ( theDatabase, "HAS_PRACTICED", "HISTORICAL_FIGURE", "RELIGION",
CC_MANY_TO_MANY, "The religion practiced by a historical figure.  A person may have
practiced several religions") ;
```

```
  NewRelation ( theDatabase, "WORKED_IN", "HISTORICAL_FIGURE", "CITY",
CC_MANY_TO_ONE, "The modern country within which modern borders the person was
working") ;
  NewRelation ( theDatabase, "BORN_IN", "HISTORICAL_FIGURE", "CITY",
CC_MANY_TO_ONE, "The modern country within which modern borders the person was born")
;
  NewRelation ( theDatabase, "BORN_IN_COUNTRY", "HISTORICAL_FIGURE",
"COUNTRY", CC_MANY_TO_ONE, "The country where the historical figure was born") ;
  NewRelation ( theDatabase, "WORKED_IN_COUNTRY", "HISTORICAL_FIGURE",
"COUNTRY", CC_MANY_TO_ONE, "The country were the Historical Figure worked in") ;
  NewRelation ( theDatabase, "IS_PART_OF", "REGION", "REGION", CC_MANY_TO_ONE,
"") ;
  NewRelation ( theDatabase, "BELONGS_TO", "CITY", "COUNTRY", CC_MANY_TO_ONE,
"") ;
  NewRelation ( theDatabase, "HAD_POPULATION", "COUNTRY", "POPULATION",
CC_ONE_TO_MANY, "Population record of the Country") ;
  NewRelation ( theDatabase, "IS_IN", "COUNTRY", "REGION", CC_MANY_TO_MANY, "")
;
  NewRelation ( theDatabase, "IN", "NOBEL_PRIZE", "NOBEL_PRIZE_FIELD",
CC_MANY_TO_ONE, "The field in which the Nobel Prize was won") ;
  NewRelation ( theDatabase, "WON_BY", "NOBEL_PRIZE", "HISTORICAL_FIGURE",
CC_MANY_TO_MANY, "The historical figure who won the Nobel Prize") ;
  NewRelation ( theDatabase, "TO", "REFERENCE", "HISTORICAL_FIGURE",
CC_MANY_TO_ONE, "The historical figure who is referenced") ;
  NewRelation ( theDatabase, "IN_BOOK", "REFERENCE", "BOOK", CC_MANY_TO_ONE,
"The book in which the reference was found.") ;
  NewRelation ( theDatabase, "FIRST_AUTHOR", "BOOK", "PERSON",
CC_MANY_TO_ONE, "The main person who wrote the book") ;
  NewRelation ( theDatabase, "OTHER_AUTHORS", "BOOK", "PERSON",
CC_MANY_TO_MANY, "The co-authors of the book, other than the first author.") ;
  NewRelation ( theDatabase, "PUBLISHED_BY", "BOOK", "PUBLISHER",
CC_MANY_TO_ONE, "The publisher of the book, if known") ;

theDatabase -> Transaction_End ( ) ;
}


void main()
{
TDataBase * DB = CreateDataBase("president.DB");
SchEd_CreateDataBase ("president.DB", DB);
CloseDataBase(DB);
}
```

Program to load the Western Culture Database (presiload.cpp)

```
//Presiload.cpp - This program loads the President's database
//Modified 5/7/98 Raquel Ghersgorin
#include "sdb3.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void
LoadScienceData(TDataBase* DB)
{
  FILE *fp;
  char *tokenPtr;
  char string[500];
  char *V[19],temp;
  char *temporal1,*temporal2,*temporal4,*temporal5;
  int i,value,temporal3,temporal6;
  fp = fopen ("science.csv", "r");
  if (fp == NULL)
   { printf("This file does not exits");}
  while(1)
   {
      i=0;
      tokenPtr=fgets(string,500,fp);
      if (tokenPtr==NULL) {
        printf("End of File ...\n");
        break; }
      V[i]=strtok(tokenPtr,",");
      i++;
      while (i<=20)
      {
          if (i==20) break;
          tokenPtr = strtok(NULL,",");
          if (strcmp(tokenPtr, "*") != 0) {
            V[i]=tokenPtr;
          } else {
            if (i==5)
              V[i]=" ";
            else
              V[i]="";
          }
          i++;
      }

//loading the persons (historical figure) data
      do
      {
          DB -> Transaction_Begin();
          Var PERSON = DB->NewAbstract( "PERSON" );
          PERSON.Relate("PERSON::LAST_NAME", V[0] );
          PERSON.Relate("PERSON::FIRST_NAME", V[1] );
          PERSON.Relate("PERSON::MIDDLE_NAME", V[2] );
          PERSON.Relate("PERSON::SUFFIX", V[14] );
```

```
PERSON.Relate("PERSON::AUTHOR_TYPE", "" );
PERSON.Relate("PERSON::PERSON_TITLE", V[3] );
PERSON.Relate("PERSON::A_K_A", V[16] );

// Get the Person who will be a Historical Figure
SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

//Relate the Historical Figure to the data
do {
        Var HISTORICAL_FIGURE( R );
        if (HISTORICAL_FIGURE.IsNull() )
                break;
        HISTORICAL_FIGURE.Categorize( "HISTORICAL_FIGURE" );
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::REFERENCE_NUMBER",
atoi(V[4]) );
        if (V[11] != "") {
                printf("V[11]: %s\n",V[11]);
                temporal1=V[11];
                temporal1=strtok(temporal1,"-");
                printf("temporal1 now : %s\n",temporal1);
                temporal2 = strtok(NULL,"");
                printf("temporal2 now : %s\n",temporal2);
                if (strcmp(temporal2,"BC") == 0) {
                        temporal3=(atoi(temporal1)*(-1));
                } else {
                        temporal3=(atoi(temporal1));
                }
                printf("for V[11] finally temporal3 is : %d\n",temporal3);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_BIRTH",
temporal3);
        }

        if (V[12] != "") {
                printf("V[12]: %s\n",V[12]);
                temporal4=V[12];
                temporal4=strtok(temporal4,"-");
                printf("temporal4 now : %s\n",temporal4);
                temporal5 = strtok(NULL,"");
                printf("temporal5 now : %s\n",temporal5);
                if (strcmp(temporal5,"BC") == 0) {
                        temporal6=(atoi(temporal4)*(-1));
                } else {
                        temporal6=(atoi(temporal4));
                }
                printf("for V[12] finally temporal6 is : %d\n",temporal6);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_DEATH",
temporal6);
```

```
                }

                //HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::BIRTH_ORDER", 0);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::PARENT", "");
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::FUNDING", "");
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::COMMENTS", V[17] );
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::METHOD", atoi(V[13]));
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::GENDER", V[15] );
                R.Next();
        } while (TRUE);

        printf("Stored HF: %s %s %s %s %s (%s)\n",V[3],V[1],V[2],V[0],V[14],V[16]);
        printf("      No.of ref: %d, Method: %d, Gender: %s, Comments: %s\n", atoi(V[4]), atoi(V[13]),
V[15], V[17]);
        printf("      DOB: %s, DOD: %s\n",V[11],V[12]);
    } while (DB->Transaction_End() == ec_concurrency);

//loading the City data and relating it to HF and Country
do
    {
        DB -> Transaction_Begin();
        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Get the Country who has the symbol
        SetQuery Rco = RangeQuery( TheRelation( "COUNTRY::SYMBOL", DB), V[6] );

        if (V[5] != "") {
                //See if the City is already in the Database
                SetQuery Rci = RangeQuery( TheRelation( "CITY::NAME", DB), V[5] );
                Var CITY1( Rci );
                if (CITY1.IsNull())
                {
                        //Create the City object
                        Var CITY = DB->NewAbstract( "CITY" );
                        CITY.Categorize( "CITY" );
                        CITY.Relate ("CITY::NAME",V[5]);
                        printf ("Stored the City : %s\n", V[5]);

                        //Relate the Person to the City where he was born
                        Var HISTORICAL_FIGURE(R);
                        HISTORICAL_FIGURE.Relate ("HISTORICAL_FIGURE::BORN_IN",CITY);
                        printf("relating the HF: %s with the City:%s where he was born\n",V[0],V[5]);

                        //Relate the City to the Country symbol
                        Var Country(Rco);
                        CITY.Relate ("CITY::BELONGS_TO",Country);
```

```
                        printf("Relating the City: %s with the Country: %s\n",V[5],V[6]);
                } else {
                        printf("The City: %s is in the database\n",V[5]);
                }
        } else {
                Var Country(Rco);
                Var HISTORICAL_FIGURE(R);

                HISTORICAL_FIGURE.Relate ("HISTORICAL_FIGURE::BORN_IN_COUNTRY",
Country);
                printf("relating the HF: %s with the Country:%s where he was born\n",V[0],V[6]);
        }
    } while (DB->Transaction_End() == ec_concurrency);

//loading the Contribution to Area data and relating it to HF and Area of Contribution

// major Contribution to Area
if (V[9] != "") {
    do
        {
        DB -> Transaction_Begin();

        //See if this area of contribution is in the database
        SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[9]);
        Var AREA_OF_CONTRIBUTION1( A0 );
        if (AREA_OF_CONTRIBUTION1.IsNull())
        {
                Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Relate ("AREA_OF_CONTRIBUTION::NAME",V[9]);
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                printf ("Stored the Area of Contribution : %s\n", V[9]);
        }else{
                printf("The Area of Contribution: %s is in the database\n",V[9]);
        }

        // Get the Area of contribution again
        SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[9]
);
        Var AREA_OF_CONTRIBUTION( A1 );

        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Create the Contribution to Area  object
```

```
        Var CONTRIBUTION_TO_AREA = DB->NewAbstract( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::MAJOR","true");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::SCHOOL_OF","");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::PROFESSION","");

        // relating the Contribution to Area and the Area of Contribution
        printf("after comments\n");
        CONTRIBUTION_TO_AREA.Relate("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTR
IBUTION);
        printf("Relating the Area of Contribution %s with the major Contribution to Area object\n",V[9]);

        // relating the Historical Figure with the Contribution to Area
        Var HISTORICAL_FIGURE(R);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
        printf("Relating the major Contribution to Area object with the HF: %s\n",V[0]);

   } while (DB->Transaction_End() == ec_concurrency);
}

// secondary Contribution to Area
if (V[10] != "") {
   do
        {
        DB -> Transaction_Begin();
        SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[10]
);
        Var AREA_OF_CONTRIBUTION1( A0 );
        if (AREA_OF_CONTRIBUTION1.IsNull())
        {
                Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::NAME",V[10]);
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                printf ("Stored the Area of Contribution : %s\n", V[10]);
        }else {
                printf("The Area of Contribution: %s is in the database\n",V[10]);
        }

        // Get the Area of contribution again
        SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[10]
);
        Var AREA_OF_CONTRIBUTION( A1 );

        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
```

```
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Create the Contribution to Area  object
        Var CONTRIBUTION_TO_AREA = DB->NewAbstract( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::MAJOR","false");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::SCHOOL_OF","");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::PROFESSION","");

        // relating the Contribution to Area and the Area of Contribution
        CONTRIBUTION_TO_AREA.Relate("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTR
IBUTION);
        printf("Relating the Area of Contribution %s with the secondary Contribution to Area
object\n",V[10]);

        // relating the Historical Figure with the Contribution to Area
        Var HISTORICAL_FIGURE(R);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
        printf("Relating the secondary Contribution to Area object with the HF: %s\n",V[0]);

    } while (DB->Transaction_End() == ec_concurrency);
}

//loading the 1st. Nobel Prize data and relating it to HF and Nobel Prize Field
do
    {
        DB -> Transaction_Begin();
        if (V[7] != "")
        {
                //Get the Person who has the name
                SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
                SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
                SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
                SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
                SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
                SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
                SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

                //getting the nobel prize field
                SetQuery S1 = RangeQuery( TheRelation( "NOBEL_PRIZE_FIELD::NAME", DB),
V[7] );

                //creating the nobel prize object
                Var NOBEL_PRIZE = DB->NewAbstract("NOBEL_PRIZE");
                NOBEL_PRIZE.Relate("NOBEL_PRIZE::YEAR",atoi(V[8]));
                printf("Creating the Nobel Prize object with year %s\n",V[8]);
                // relating the Nobel Prize Field in Nobel Prize
                Var NOBEL_PRIZE_FIELD(S1);
                NOBEL_PRIZE.Relate("NOBEL_PRIZE::IN",NOBEL_PRIZE_FIELD);
                printf("Relating the Nobel Prize Field %s with the nobel prize of year %s\n",V[7],V[8]);
```

```
                // relating the Historical Figure with the Nobel Prize
                Var HISTORICAL_FIGURE(R);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
WON_BY",NOBEL_PRIZE);
                printf("Relating the Nobel Prize object with the HF: %s\n",V[0]);
        }
    } while (DB->Transaction_End() == ec_concurrency);

//loading the 2nd. Nobel Prize data and relating it to HF and Nobel Prize Field
do
    {
        DB -> Transaction_Begin();
        if (V[18] != "")
        {
                //Get the Person who has the name
                SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
                SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[1] );
                SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[2] );
                SetQuery R4 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[3] );
                SetQuery R5 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[14] );
                SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[16] );
                SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

                //getting the nobel prize field
                SetQuery S1 = RangeQuery( TheRelation( "NOBEL_PRIZE_FIELD::NAME", DB),
V[18] );
                //creating the nobel prize object
                Var NOBEL_PRIZE = DB->NewAbstract("NOBEL_PRIZE");
                NOBEL_PRIZE.Relate("NOBEL_PRIZE::YEAR",atoi(V[19]));
                printf("Creating the Nobel Prize object with year %s\n",V[19]);

                // relating the Nobel Prize Field in Nobel Prize
                Var NOBEL_PRIZE_FIELD(S1);
                NOBEL_PRIZE.Relate("NOBEL_PRIZE::IN",NOBEL_PRIZE_FIELD);
                printf("Relating the Nobel Prize Field %s with the nobel prize of year
%s\n",V[18],V[19]);

                // relating the Historical Figure with the Nobel Prize
                Var HISTORICAL_FIGURE(R);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
WON_BY",NOBEL_PRIZE);
                printf("Relating the Nobel Prize object with the HF: %s\n",V[0]);
        }
    } while (DB->Transaction_End() == ec_concurrency);

printf("**************** next person *****************\n");
} //endwhile for each line
}
void
LoadCountryData(TDataBase* DB)
{
  FILE *fp;
```

```
  char *tokenPtr_Country;
  char string_Country[50]=" ";
  char blanco;
  char *V_Country[2];
  int i=0;

  fp = fopen ("countries.csv", "r");
  printf("after oppening countries\n");
  if (fp == NULL)
    { printf("This file does not exits");}

  while(1)
   {
      i=0;
      tokenPtr_Country=fgets(string_Country,50,fp);
      if (tokenPtr_Country==NULL) {
        printf("End of File ...\n");
        break; }
     V_Country[i]=strtok(tokenPtr_Country,",");
      i++;
      while (i<=2)
      {
      if (i==2) break;
      tokenPtr_Country = strtok(NULL,",");
      V_Country[i]=tokenPtr_Country;
      i++;
      }
      do
      {
           DB -> Transaction_Begin();
           Var COUNTRY = DB->NewAbstract("COUNTRY");
           COUNTRY.Relate("COUNTRY::SYMBOL",V_Country[1]);
           COUNTRY.Relate ("COUNTRY::NAME",V_Country[0]);
           printf("Storing : %s %s\n",V_Country[0],V_Country[1]);
       } while (DB->Transaction_End() == ec_concurrency);
   }
   }


void
LoadNobelPrizeFieldData (TDataBase* DB)
{
   FILE *fp;
   char *tokenPtr_NPF;
   char string_NPF[50]=" ";
   char *V_NPF[2];
   int i=0;
   fp = fopen ("nobel_prize_fields.csv", "r");
   if (fp == NULL)
     { printf("This file does not exits");}
   while(1)
    {
       i=0;
```

```
      tokenPtr_NPF=fgets(string_NPF,50,fp);
      if (tokenPtr_NPF==NULL) {
       printf("End of File ...\n");
       break; }
      V_NPF[i]=strtok(tokenPtr_NPF,",");
      i++;
      while (i<=2)
      {
      if (i==2) break;
      tokenPtr_NPF = strtok(NULL,",");
      V_NPF[i]=tokenPtr_NPF;
      i++;
      }
      do
      {
          DB -> Transaction_Begin();
          Var NOBEL_PRIZE_FIELD = DB->NewAbstract("NOBEL_PRIZE_FIELD");
          NOBEL_PRIZE_FIELD.Relate ("NOBEL_PRIZE_FIELD::NAME",V_NPF[0]);
          printf("Storing : %s\n",V_NPF[0]);
      } while (DB->Transaction_End() == ec_concurrency);
  }
}

void
LoadArtData(TDataBase* DB)
{
  FILE *fp;
  char *tokenPtr;
  char *temporal1,*temporal2,*temporal4,*temporal5;
  char string[500];
  char *V[20],temp;
  int i,value,temporal3,temporal6;

  printf("BEFORE opening file\n");
  fp = fopen ("artists.csv", "r");
  if (fp == NULL)
   { printf("This file does not exits");}
  while(1)
   {
      i=0;
      tokenPtr=fgets(string,500,fp);
      if (tokenPtr==NULL) {
       printf("End of File ...\n");
       break; }
      V[i]=strtok(tokenPtr,",");
      i++;
      while (i<=21)
      {
          if (i==21) break;
          tokenPtr = strtok(NULL,",");
          if (strcmp(tokenPtr, "*") != 0)
            {
                          V[i]=tokenPtr;
```

```
        }
        else
        {
                V[i]="";
        }
        i++;
    }

//loading the persons (historical figure) data
    do
    {
        DB -> Transaction_Begin();

        Var PERSON = DB->NewAbstract( "PERSON" );
        PERSON.Relate("PERSON::LAST_NAME", V[0] );
        PERSON.Relate("PERSON::FIRST_NAME", V[2] );
        PERSON.Relate("PERSON::MIDDLE_NAME", V[3] );
        PERSON.Relate("PERSON::SUFFIX", V[5] );
        PERSON.Relate("PERSON::AUTHOR_TYPE", "" );
        PERSON.Relate("PERSON::PERSON_TITLE", V[6] );
        PERSON.Relate("PERSON::A_K_A", V[7] );

        // Get the Person who will be a Historical Figure
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Relate the Historical Figure to the data
        do {
                Var HISTORICAL_FIGURE( R );
                if (HISTORICAL_FIGURE.IsNull() )
                        break;
                HISTORICAL_FIGURE.Categorize( "HISTORICAL_FIGURE" );
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::REFERENCE_NUMBER",
atoi(V[9]) );
                if (V[15] != "") {
                        printf("V[15]: %s\n",V[15]);
                        temporal1=V[15];
                        temporal1=strtok(temporal1,"-");
                        printf("temporal1 now : %s\n",temporal1);
                        temporal2 = strtok(NULL,"");
                        printf("temporal2 now : %s\n",temporal2);
                        if (strcmp(temporal2,"BC") == 0) {
                                temporal3=(atoi(temporal1)*(-1));
                        } else {
                                temporal3=(atoi(temporal1));
                        }
                        printf("for V[15] finally temporal3 is : %d\n",temporal3);
```

```
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_BIRTH",
temporal3);
                }

                if (V[16] != "") {
                        printf("V[16]: %s\n",V[16]);
                        temporal4=V[16];
                        temporal4=strtok(temporal4,"-");
                        printf("temporal4 now : %s\n",temporal4);
                        temporal5 = strtok(NULL,"");
                        printf("temporal5 now : %s\n",temporal5);
                        if (strcmp(temporal5,"BC") == 0) {
                                temporal6=(atoi(temporal4)*(-1));
                        } else {
                                temporal6=(atoi(temporal4));
                        }
                        printf("for V[16] finally temporal6 is : %d\n",temporal6);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_DEATH",
temporal6);
                }

                //HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::BIRTH_ORDER", 0);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::PARENT", "");
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::FUNDING", "");
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::COMMENTS", V[8]);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::METHOD", atoi(V[12]));
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::GENDER", V[4] );
                R.Next();
        } while (TRUE);

        printf("Stored HF: %s %s %s %s\n",V[6],V[2],V[3],V[0],V[5]);
        printf("also known as: %s\n",V[7]);
        printf("      No.of ref: %d, Method: %d, Gender: %s, Comments: %s\n", atoi(V[9]), atoi(V[12]),
V[4], V[8]);
        printf("      DOB: %s, DOD: %s\n",V[15],V[16]);
    } while (DB->Transaction_End() == ec_concurrency);

//loading the City data and relating it to HF and Country
    do
        {
        DB -> Transaction_Begin();
        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Get the Country who has the symbol
        SetQuery Rco = RangeQuery( TheRelation( "COUNTRY::SYMBOL", DB), V[14] );
        Var Country(Rco);
```

```
        if (V[13] != "") {
                //See if the City is already in the Database
                SetQuery Rci = RangeQuery( TheRelation( "CITY::NAME", DB), V[13] );
                Var CITY1( Rci );
                if (CITY1.IsNull())
                {
                        //Create the City object
                        Var CITY = DB->NewAbstract( "CITY" );
                        CITY.Categorize( "CITY" );
                        CITY.Relate ("CITY::NAME",V[13]);
                        printf ("Stored the City : %s\n", V[13]);

                        //Relate the Person to the City where he was born
                        Var HISTORICAL_FIGURE(R);
                        HISTORICAL_FIGURE.Relate ("HISTORICAL_FIGURE::BORN_IN",CITY);
                        printf("relating the HF: %s with the City:%s where he was born\n",V[0],V[13]);

                        //Relate the City to the Country symbol
                        if (V[14] != "") {
                                CITY.Relate ("CITY::BELONGS_TO",Country);
                                printf("Relating the City: %s with the Country: %s\n",V[13],V[14]);
                        }
                }
                else
                {
                        printf("The City: %s is in the database\n",V[13]);
                }
        }
        else
        {
                if (V[14] != "") {
                        Var HISTORICAL_FIGURE(R);
                        HISTORICAL_FIGURE.Relate
("HISTORICAL_FIGURE::BORN_IN_COUNTRY", Country);
                        printf("relating the HF: %s with the Country:%s where he was
born\n",V[0],V[14]);
                }
        }

    } while (DB->Transaction_End() == ec_concurrency);

//loading the Contribution to Area data and relating it to HF and Area of Contribution

// major Contribution to Area
if (V[17] != "") {
    do
        {
        DB -> Transaction_Begin();

        //See if this area of contribution is in the database
        SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[17]
);
        Var AREA_OF_CONTRIBUTION1( A0 );
```

```
        if (AREA_OF_CONTRIBUTION1.IsNull())
        {
                Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::NAME",V[17]);
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                printf ("Stored the Area of Contribution : %s\n", V[17]);

        }
        else
        {
                printf("The Area of Contribution: %s is in the database\n",V[17]);
        }

        // Get the Area of contribution again
        SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[17]
);
        Var AREA_OF_CONTRIBUTION( A1 );

        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Create the Contribution to Area  object
        Var CONTRIBUTION_TO_AREA = DB->NewAbstract( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::MAJOR","true");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::SCHOOL_OF",V[19]);
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::PROFESSION","");

        // relating the Contribution to Area and the Area of Contribution
        printf("after comments\n");
        CONTRIBUTION_TO_AREA.Relate("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTR
IBUTION);
        printf("Relating the Area of Contribution %s with the major Contribution to Area object\n",V[17]);

        // relating the Historical Figure with the Contribution to Area
        Var HISTORICAL_FIGURE(R);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
        printf("Relating the major Contribution to Area object with the HF: %s\n",V[0]);

        } while (DB->Transaction_End() == ec_concurrency);
}
```

```
// secondary Contribution to Area
if (V[18] != "") {
    do
        {
        DB -> Transaction_Begin();
        SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[18]
);
        Var AREA_OF_CONTRIBUTION1( A0 );
        if (AREA_OF_CONTRIBUTION1.IsNull())
        {
                Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::NAME",V[18]);
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                printf ("Stored the Area of Contribution : %s\n", V[18]);
        }
        else
        {
                printf("The Area of Contribution: %s is in the database\n",V[18]);
        }

        // Get the Area of contribution again
        SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[18]
);
        Var AREA_OF_CONTRIBUTION( A1 );

        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Create the Contribution to Area  object
        Var CONTRIBUTION_TO_AREA = DB->NewAbstract( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::MAJOR","false");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::SCHOOL_OF",V[20]);
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::PROFESSION","");

        // relating the Contribution to Area and the Area of Contribution
        CONTRIBUTION_TO_AREA.Relate("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTR
IBUTION);
        printf("Relating the Area of Contribution %s with the secondary Contribution to Area
object\n",V[18]);

        // relating the Historical Figure with the Contribution to Area
```

```
        Var HISTORICAL_FIGURE(R);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
        printf("Relating the secondary Contribution to Area object with the HF: %s\n",V[0]);
} while (DB->Transaction_End() == ec_concurrency);
}

printf("**************** next person *****************\n");

} //endwhile for each line
}



void
LoadPhiloData(TDataBase* DB)
{
  FILE *fp;
  char *tokenPtr;
  char *temporal1,*temporal2,*temporal4,*temporal5;
  char string[500];
  char *V[20],temp;
  int i,value,temporal3,temporal6;

  printf("BEFORE opening file\n");
  fp = fopen ("philosophy.csv", "r");
  if (fp == NULL)
   { printf("This file does not exits");}
  while(1)
   {
     i=0;
     tokenPtr=fgets(string,500,fp);
     if (tokenPtr==NULL) {
       printf("End of File ...\n");
       break; }
     V[i]=strtok(tokenPtr,",");
     i++;
     while (i<=21)
     {
        if (i==21) break;
        tokenPtr = strtok(NULL,",");
        if (strcmp(tokenPtr, "*") != 0)
          {
                V[i]=tokenPtr;
          }
        else
          {
                 V[i]="";
          }
       i++;
     }

 //loading the persons (historical figure) data
```

```
do
{
    DB -> Transaction_Begin();

    Var PERSON = DB->NewAbstract( "PERSON" );
    PERSON.Relate("PERSON::LAST_NAME", V[0] );
    PERSON.Relate("PERSON::FIRST_NAME", V[2] );
    PERSON.Relate("PERSON::MIDDLE_NAME", V[3] );
    PERSON.Relate("PERSON::SUFFIX", V[5] );
    PERSON.Relate("PERSON::AUTHOR_TYPE", "" );
    PERSON.Relate("PERSON::PERSON_TITLE", V[6] );
    PERSON.Relate("PERSON::A_K_A", V[7] );

    // Get the Person who will be a Historical Figure
    SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
    SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
    SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
    SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
    SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
    SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
    SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

    //Relate the Historical Figure to the data
    do {
        Var HISTORICAL_FIGURE( R );
        if (HISTORICAL_FIGURE.IsNull() )
        break;
        HISTORICAL_FIGURE.Categorize( "HISTORICAL_FIGURE" );
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::REFERENCE_NUMBER",
atoi(V[9]) );
        if (V[15] != "") {
            printf("V[15]: %s\n",V[15]);
            temporal1=V[15];
            temporal1=strtok(temporal1,"-");
            printf("temporal1 now : %s\n",temporal1);
            temporal2 = strtok(NULL,"");
            printf("temporal2 now : %s\n",temporal2);
            if (strcmp(temporal2,"BC") == 0) {
                temporal3=(atoi(temporal1)*(-1));
            } else {
                temporal3=(atoi(temporal1));
            }
            printf("for V[15] finally temporal3 is : %d\n",temporal3);

        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_BIRTH",
temporal3);
        }

        if (V[16] != "") {
            printf("V[16]: %s\n",V[16]);
            temporal4=V[16];
            temporal4=strtok(temporal4,"-");
            printf("temporal4 now : %s\n",temporal4);
```

```
                        temporal5 = strtok(NULL,"");
                        printf("temporal5 now : %s\n",temporal5);
                        if (strcmp(temporal5,"BC") == 0) {
                                temporal6=(atoi(temporal4)*(-1));
                        } else {
                                temporal6=(atoi(temporal4));
                        }
                        printf("for V[16] finally temporal6 is : %d\n",temporal6);

        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::DATE_OF_DEATH", temporal6);
                }

        //HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::BIRTH_ORDER", 0);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::PARENT", "");
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::FUNDING", "");
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::COMMENTS", V[8]);
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::METHOD", atoi(V[12]));
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::GENDER", V[4] );
        R.Next();
        } while (TRUE);

        printf("Stored HF: %s %s %s %s\n",V[6],V[2],V[3],V[0],V[5]);
        printf("also known as: %s\n",V[7]);
        printf("      No.of ref: %d, Method: %d, Gender: %s, Comments: %s\n", atoi(V[9]), atoi(V[12]),
V[4], V[8]);
        printf("      DOB: %s, DOD: %s\n",V[15],V[16]);
    } while (DB->Transaction_End() == ec_concurrency);

//loading the City data and relating it to HF and Country
    do
        {
                DB -> Transaction_Begin();
                //Get the Person who has the name
                SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
                SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
                SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
                SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
                SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
                SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
                SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

                //Get the Country who has the symbol
                SetQuery Rco = RangeQuery( TheRelation( "COUNTRY::SYMBOL", DB), V[14] );
                Var Country(Rco);
                if (V[13] != "") {
                        //See if the City is already in the Database
                        SetQuery Rci = RangeQuery( TheRelation( "CITY::NAME", DB), V[13] );
                        Var CITY1( Rci );
                        if (CITY1.IsNull())
                        {
                                //Create the City object
                                Var CITY = DB->NewAbstract( "CITY" );
                                CITY.Categorize( "CITY" );
```

```
                                CITY.Relate ("CITY::NAME",V[13]);
                                printf ("Stored the City : %s\n", V[13]);

                                //Relate the Person to the City where he was born
                                Var HISTORICAL_FIGURE(R);

                                HISTORICAL_FIGURE.Relate
("HISTORICAL_FIGURE::BORN_IN",CITY);
                                printf("relating the HF: %s with the City:%s where he was
born\n",V[0],V[13]);

                                //Relate the City to the Country symbol
                                if (V[14] != "") {
                                        CITY.Relate ("CITY::BELONGS_TO",Country);
                                        printf("Relating the City: %s with the Country:
%s\n",V[13],V[14]);
                                }
                        }
                        else
                        {
                                printf("The City: %s is in the database\n",V[13]);
                        }
                }
                else
                {
                        if (V[14] != "") {
                                Var HISTORICAL_FIGURE(R);
                                HISTORICAL_FIGURE.Relate
("HISTORICAL_FIGURE::BORN_IN_COUNTRY",Country);
                                printf("relating the HF: %s with the Country:%s where he was
born\n",V[0],V[14]);
                        }
                }

        } while (DB->Transaction_End() == ec_concurrency);

//loading the Contribution to Area data and relating it to HF and Area of Contribution

// major Contribution to Area
if (V[17] != "") {
        do
        {
                DB -> Transaction_Begin();

                //See if this area of contribution is in the database
                SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME",
DB), V[17] );
                Var AREA_OF_CONTRIBUTION1( A0 );
                if (AREA_OF_CONTRIBUTION1.IsNull())
                {
                        Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                        AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
```

```
                        AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::NAME",V[17]);
                        AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                        printf ("Stored the Area of Contribution : %s\n", V[17]);

                }
                else
                {
                        printf("The Area of Contribution: %s is in the database\n",V[17]);
                }

                // Get the Area of contribution again
                SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME",
DB), V[17] );
                Var AREA_OF_CONTRIBUTION( A1 );

                //Get the Person who has the name
                SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
                SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
                SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
                SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
                SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
                SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
                SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

                //Create the Contribution to Area  object
                Var CONTRIBUTION_TO_AREA = DB->NewAbstract(
"CONTRIBUTION_TO_AREA" );
                CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
                CONTRIBUTION_TO_AREA.Relate
("CONTRIBUTION_TO_AREA::MAJOR","true");
                CONTRIBUTION_TO_AREA.Relate
("CONTRIBUTION_TO_AREA::SCHOOL_OF",V[19]);
                CONTRIBUTION_TO_AREA.Relate
("CONTRIBUTION_TO_AREA::PROFESSION","");

                // relating the Contribution to Area and the Area of Contribution
                printf("after comments\n");
                CONTRIBUTION_TO_AREA.Relate
("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTRIBUTION);
                printf("Relating the Area of Contribution %s with the major Contribution to Area
object\n",V[17]);

                // relating the Historical Figure with the Contribution to Area
                Var HISTORICAL_FIGURE(R);
                HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
                printf("Relating the major Contribution to Area object with the HF: %s\n",V[0]);
        } while (DB->Transaction_End() == ec_concurrency);
}
```

```
// secondary Contribution to Area
if (V[18] != "") {
    do
        {
        DB -> Transaction_Begin();
        SetQuery A0 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[18]
);
        Var AREA_OF_CONTRIBUTION1( A0 );
        if (AREA_OF_CONTRIBUTION1.IsNull())
        {
                Var AREA_OF_CONTRIBUTION1 = DB->NewAbstract(
"AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Categorize( "AREA_OF_CONTRIBUTION" );
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::NAME",V[18]);
                AREA_OF_CONTRIBUTION1.Relate
("AREA_OF_CONTRIBUTION::COMMENTS"," ");
                printf ("Stored the Area of Contribution : %s\n", V[18]);
        }
        else
        {
                printf("The Area of Contribution: %s is in the database\n",V[18]);
        }

        // Get the Area of contribution again
        SetQuery A1 = RangeQuery( TheRelation( "AREA_OF_CONTRIBUTION::NAME", DB), V[18]
);
        Var AREA_OF_CONTRIBUTION( A1 );

        //Get the Person who has the name
        SetQuery R1 = RangeQuery( TheRelation( "PERSON::LAST_NAME", DB), V[0] );
        SetQuery R2 = RangeQuery( TheRelation( "PERSON::FIRST_NAME", DB), V[2] );
        SetQuery R3 = RangeQuery( TheRelation( "PERSON::MIDDLE_NAME", DB), V[3] );
        SetQuery R4 = RangeQuery( TheRelation( "PERSON::SUFFIX", DB), V[5] );
        SetQuery R5 = RangeQuery( TheRelation( "PERSON::PERSON_TITLE", DB), V[6] );
        SetQuery R6 = RangeQuery( TheRelation( "PERSON::A_K_A", DB), V[7] );
        SetAnd R = R1 & R2 & R3 & R4 & R5 & R6;

        //Create the Contribution to Area  object
        Var CONTRIBUTION_TO_AREA = DB->NewAbstract( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Categorize( "CONTRIBUTION_TO_AREA" );
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::MAJOR","false");
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::SCHOOL_OF",V[20]);
        CONTRIBUTION_TO_AREA.Relate ("CONTRIBUTION_TO_AREA::PROFESSION","");

        // relating the Contribution to Area and the Area of Contribution
        CONTRIBUTION_TO_AREA.Relate("CONTRIBUTION_TO_AREA::OF",AREA_OF_CONTR
IBUTION);
        printf("Relating the Area of Contribution %s with the secondary Contribution to Area
object\n",V[18]);

        // relating the Historical Figure with the Contribution to Area
        Var HISTORICAL_FIGURE(R);
```

```
        HISTORICAL_FIGURE.Relate("HISTORICAL_FIGURE::-
BY",CONTRIBUTION_TO_AREA);
        printf("Relating the secondary Contribution to Area object with the HF: %s\n",V[0]);

   } while (DB->Transaction_End() == ec_concurrency);
}

printf("**************** next person *****************\n");

} //endwhile for each line
}


void
LoadGeneralTables(TDataBase* DB)
{
  LoadCountryData(DB);
  LoadNobelPrizeFieldData(DB);
 }


int main() {
 TDataBase* DB = OpenDataBase("president.DB");
 printf("BEFORE LOADING DATA\n");
 LoadGeneralTables(DB);
 LoadScienceData(DB);
 LoadArtData(DB);
 LoadPhiloData(DB);
 CloseDataBase(DB);
 printf(" DONE ...\n");
}
```

Steps to create and load the Western Culture's Database

The Western Culture's database is created and loaded using the 6th. version of the SDB (binary engine) on NT. The Files are located in N12, on the directory InetPub\wwwroot\president\database\

To create the database:
- Open the project called database.dsw using Microsoft Visual C++ (Developer Studio).
- On the source files option, add the file called "presicreate.cpp", which is the file to create the database.
- Build the project, and it will generate a file called database.exe, located in the directory InetPub\wwwroot\president\database\release. Copy this file in the directory InetPub\wwwroot\president\database\ as presicreate.exe.
- Now, execute presicreate.exe, and it will generate the directory president.db (the semantic database).

To load the database:
- Open the project called database.dsw using Microsoft Visual C++ (Developer Studio).
- On the source files option, add the file called "presiload.cpp", which is the file to load the database. This C++ program takes the data from: countries.csv, artists.csv, nobel_prize_fields.csv, science.csv., and philosophy.csv.
- Build the project, and it will generate a file called database.exe, located in the directory InetPub\wwwroot\president\database\release. Copy this file in the directory InetPub\wwwroot\president\database\ as presiload.exe.
- Now, execute presiload.exe, and it will load president.db (the semantic database).

APPENDIX F (Semantic web-interface appendix)

Flow Chart of the Implementation of

"Western Culture Semantic Database Application" – Web Interface

wwwroot/president/index.html :
    wwwroot/president/main_menu.html -> ...
    wwwroot/president/presentation.html -> wwwroot/president/back.gif

Main Menu, wwwroot/president/main_menu.html :
Primary Forms :
    Region: wwwroot/president/regionform.html ->
        wwwroot/president/regionupform.html ->
            wwwroot/president/macrofile/regioni.mac -> wwwroot/president/macrofile/regionr.mac
            wwwroot/president/macrofile/listregions.mac ->
    wwwroot/president/macrofile/region_selected_for_region.mac
        wwwroot/president/macrofile/upregion.mac ->
            wwwroot/president/macrofile/modregion.mac ->
                wwwroot/president/macrofile/regionm.mac
                wwwroot/president/macrofile/regiond.mac
    Country: wwwroot/president/countryform.html ->
        wwwroot/president/countryupform.html ->
            wwwroot/president/macrofile/countryi.mac -> wwwroot/president/macrofile/countryr.mac
            wwwroot/president/macrofile/listregions_for_country.mac ->
                wwwroot/president/macrofile/region_selected_fo
                r_country.mac
        wwwroot/president/macrofile/upcountry.mac ->
            wwwroot/president/macrofile/modcountry.mac ->

wwwroot/president/macrofile/countrym.mac
wwwroot/president/macrofile/countryd.mac

City: wwwroot/president/cityform.html ->
wwwroot/president/cityupform.html ->
wwwroot/president/macrofile/cityi.mac -> wwwroot/president/macrofile/cityr.mac
wwwroot/president/macrofile/listcountry_for_city.mac ->
wwwroot/president/macrofile/country_selected_for_city.mac
wwwroot/president/macrofile/upcity.mac ->
wwwroot/president/macrofile/modcity.mac ->
wwwroot/president/macrofile/citym.mac
wwwroot/president/macrofile/cityd.mac

Religion: wwwroot/president/religionform.html ->
wwwroot/president/macrofile/religioni.mac -> wwwroot/president/macrofile/religionr.mac ->
wwwroot/president/macrofile/upreligion.mac ->
wwwroot/president/macrofile/modreligion.mac ->
wwwroot/president/macrofile/religionm.mac
wwwroot/president/macrofile/religiond.mac

Nobel Prize Field: wwwroot/president/nobelprizefieldform.html ->
wwwroot/president/macrofile/nobelprizefieldi.mac -> wwwroot/president/macrofile/nobelprizefieldr.mac ->
wwwroot/president/macrofile/upnobelprizefield.mac ->
wwwroot/president/macrofile/modnobelprizefield.mac ->
wwwroot/president/macrofile/nobelprizefieldm.mac
wwwroot/president/macrofile/nobelprizefieldd.mac

Area of Contribution: wwwroot/president/areaofcontributionform.html ->
wwwroot/president/areaofcontributionupform.html ->
wwwroot/president/macrofile/areaofcontributioni.mac ->
wwwroot/president/macrofile/areaofcontributionr.mac
wwwroot/president/macrofile/listareasofcontribution.mac ->

wwwroot/president/macrofile/
areaofcontribution_selected_for_aoc.mac
wwwroot/president/macrofile/upareaofcontribution.mac ->
wwwroot/president/macrofile/modareaofcontribution.mac ->
wwwroot/president/macrofile/areaofcontributionm.mac
wwwroot/president/macrofile/areaofcontributiond.mac

Population: wwwroot/president/populationform.html ->
wwwroot/president/macrofile/populationi.mac -> wwwroot/president/macrofile/populationr.mac ->
wwwroot/president/macrofile/uppopulation.mac ->
wwwroot/president/macrofile/modpopulation.mac ->
wwwroot/president/macrofile/populationm.mac
wwwroot/president/macrofile/populationd.mac

Personal Forms :

Person: wwwroot/president/personform.html ->
wwwroot/president/macrofile/personi.mac -> wwwroot/president/macrofile/personr.mac ->
wwwroot/president/macrofile/upperson.mac ->
wwwroot/president/macrofile/modperson.mac ->
wwwroot/president/macrofile/personm.mac
wwwroot/president/macrofile/persond.mac

Historical Figure: wwwroot/president/historicalfigureform.html ->
wwwroot/president/hfupform.html ->
wwwroot/president/macrofile/historicalfigurei.mac ->
wwwroot/president/macrofile/historicalfigurer.mac
wwwroot/president/macrofile/hf_list_for_selections.mac ->
wwwroot/president/macrofile/hf_selections.mac
wwwroot/president/macrofile/hf_list_for_selections_person.mac->
wwwroot/president/macrofile/hf_selections.mac
wwwroot/president/macrofile /hf_year_npf.mac ->
wwwroot/president/macrofile /hf_list_for_selections_npf.mac ->
wwwroot/president/macrofile/hf_selections_npf.mac

wwwroot/president/macrofile /hf_data_aoc.mac ->
wwwroot/president/macrofile/hf_list_for_selections_aoc.mac ->
wwwroot/president/macrofile/hf_selections_aoc.mac
wwwroot/president/macrofile/hf_selections.mac
wwwroot/president/macrofile/uphistoricalfigure.mac ->
wwwroot/president/macrofile/modhistoricalfigure.mac ->
wwwroot/president/macrofile/historicalfigurem.mac
wwwroot/president/macrofile/historicalfigured.mac
wwwroot/president/macrofile/hf_list_for_selections.mac ->
wwwroot/president/macrofile/hf_selections.mac
wwwroot/president/macrofile/hf_list_for_selections_person.mac
wwwroot/president/macrofile /hf_year_npf.mac ->
wwwroot/president/macrofile /hf_list_for_selections_npf.mac ->
wwwroot/president/macrofile/hf_selections_npf.mac
wwwroot/president/macrofile /hf_data_aoc.mac ->
wwwroot/president/macrofile/hf_list_for_selections_aoc.mac ->
wwwroot/president/macrofile/hf_selections_aoc.mac
wwwroot/president/macrofile/hf_but_for_prev_sele_per.mac ->
wwwroot/president/macrofile/list_prev_per.mac ->
wwwroot/president/macrofile/hf_remove_sel_per.mac
wwwroot/president/macrofile/hf_but_for_prev_sele_rel.mac ->
wwwroot/president/macrofile/list_prev_rel.mac ->
wwwroot/president/macrofile/hf_remove_sel_rel.mac
wwwroot/president/macrofile/hf_but_for_prev_sele_npf.mac ->
wwwroot/president/macrofile/list_prev_npf.mac ->
wwwroot/president/macrofile/hf_remove_sel_npf.mac
wwwroot/president/macrofile/hf_but_for_prev_sele_aoc.mac ->
wwwroot/president/macrofile/list_prev_aoc.mac ->
wwwroot/president/macrofile/hf_remove_sel_aoc.mac

Relation Forms :

Religion, Country and Period: wwwroot/president/rel_coun_per_form.html ->
wwwroot/president/rel_coun_per_upform.html ->
wwwroot/president/macrofile/rel_coun_peri.mac ->
wwwroot/president/macrofile/rel_coun_perr.mac
wwwroot/president/rel_coun_per_down.html ->
wwwroot/president/macrofile/listreligion_for_period.mac ->
wwwroot/president/macrofile/rel_selected.mac
wwwroot/president/macrofile/listcountry_for_period.mac ->
wwwroot/president/macrofile/coun_selected.mac
wwwroot/president/macrofile/up_rel_coun_per.mac ->
wwwroot/president/macrofile/mod_rel_coun_per.mac ->
wwwroot/president/macrofile/rel_coun_perm.mac
wwwroot/president/macrofile/rel_coun_perd.mac

Contribution to Area and Period: wwwroot/president/cont_area_per_form.html ->
wwwroot/president/cont_area_per_upform.html ->
wwwroot/president/macrofile/cont_area_peri.mac ->
wwwroot/president/macrofile/cont_area_perr.mac
wwwroot/president/cont_area_per_down.html ->
wwwroot/president/macrofile/listcontribution_for_period.mac ->
wwwroot/president/macrofile/rel_selected.mac
wwwroot/president/macrofile/up_cont_area_per.mac ->
wwwroot/president/macrofile/mod_cont_area_per.mac ->
wwwroot/president/macrofile/cont_area_perm.mac
wwwroot/president/macrofile/cont_area_perd.mac

Country and Population: wwwroot/president/coun_popu_form.html ->
wwwroot/president/coun_popu_upform.html ->
wwwroot/president/macrofile/coun_popui.mac -> wwwroot/president/macrofile/coun_popur.mac
wwwroot/president/macrofile/listcountry_for_popu.mac ->
wwwroot/president/macrofile/country_selected_for_popu.mac

wwwroot/president/macrofile/up_coun_popu.mac ->
wwwroot/president/macrofile/mod_coun_popu.mac ->
wwwroot/president/macrofile/coun_popum.mac
wwwroot/president/macrofile/coun_popud.mac

General Reports :

wwwroot/president/macrofile/repcity.mac
wwwroot/president/macrofile/repperson.mac
wwwroot/president/macrofile/rephistoricalfigure.mac -> wwwroot/president/macrofile/rephf.mac
wwwroot/president/macrofile/repcountry.mac
wwwroot/president/macrofile/repnobelprizefield.mac
wwwroot/president/macrofile/repareaofcontribution.mac

HF Reports and Graphs:

wwwroot/president/rephf_city_form.html ->
wwwroot/president/macrofile/rephf_cityi.mac ->
wwwroot/president/macrofile/rephf1_display_city.mac
wwwroot/president/macrofile/rephf2_display_city.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac
wwwroot/president/rephf_country_form.html ->
wwwroot/president/macrofile/rephf_countryi.mac ->
wwwroot/president/macrofile/rephf1_display_country.mac
wwwroot/president/macrofile/rephf2_display_country.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac
wwwroot/president/rephf_country_city_form.html ->
wwwroot/president/macrofile/rephf_country_cityi.mac ->
wwwroot/president/macrofile/rephf1_display_country_city.mac
wwwroot/president/macrofile/rephf2_display_country_city.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac
wwwroot/president/rephf_aoc_form.html ->
wwwroot/president/macrofile/rephf_aoci.mac ->
wwwroot/president/macrofile/rephf1_display_aoc.mac

wwwroot/president/macrofile/rephf2_display_aoc.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac
wwwroot/president/rephf_nobelp_form.html ->
wwwroot/president/macrofile/rephf_nobelpi.mac ->
wwwroot/president/macrofile/rephf1_display_nobelp.mac
wwwroot/president/macrofile/rephf2_display_nobelp.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac
wwwroot/president/rephf_title_form.html ->
wwwroot/president/macrofile/rephf_titlei.mac ->
wwwroot/president/macrofile/rephf1_display_title.mac
wwwroot/president/macrofile/rephf2_display_title.mac
wwwroot/president/rephf_year_form.html ->
wwwroot/president/macrofile/rephf_yeari.mac ->
wwwroot/president/macrofile/rephf1_display_year.mac
wwwroot/president/macrofile/rephf2_display_year.mac
wwwroot/president/rephf_span_form.html ->
wwwroot/president/macrofile/rephf_spani.mac ->
wwwroot/president/macrofile/rephf_display_span.mac
wwwroot/president/rephf_country_span_form.html ->
wwwroot/president/macrofile/rephf_country_spani.mac -> wwwroot/president/macrofile/
rephf_display_country_span.mac
wwwroot/president/macrofile/butt_list_report.mac -> wwwroot/president/macrofile/listreport.mac

Advanced Ad-Hoc Retrieving Tools :

wwwroot/sdbbrowser.exe
wwwroot/sdbnavigator.exe

Some programs: Web Interface implementation.

wwwroot/president/main_menu.html

```
<html>
<head>
<BASE target="operations">
<TITLE> Page of Options </TITLE>
</head>

<body bgcolor="#CCCCFF" text="#000000" link="#0000FF"
   alink="#FF2020" vlink="#800020">

<center>

<font size=5 color="green"><B> Options </B></font>
<HR SIZE=4 WIDTH="60%">

<TABLE border=0 width=100% valign="bottom" align="left">

<TR><TD align="center" COLSPAN=2><B><FONT size=+1 color="navy">Input
Forms</FONT></B><bR>
<HR SIZE=4 WIDTH="100%"></TD></TR>

<TR><TD><B><FONT size=+1 color="green">General Forms: </FONT></B></TD></TR>

<TR><TD ALIGN="right"><font color="black">

<FORM ACTION="http://www.fiu.edu/cgi-bin/gourl" METHOD="GET">
<B><SELECT NAME="URL">
<OPTION VALUE="http://n12.cs.fiu.edu/president/regionform.html">Region</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/countryform.html">Country</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/cityform.html">City</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/religionform.html">Religion</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/nobelprizefieldform.html">Nobel Prize
Field</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/areaofcontributionform.html">Area of
Contribution</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/populationform.html">World Population</OPTION>
</SELECT></TD>
<TD><INPUT type="submit"  value="Go">
</TD></TR></font>
</FORM>

<TR><TD><B><FONT size=+1 color="green">Personal Forms: </FONT></B></TD></TR>

<TR><TD ALIGN="right"><font color="black">

<FORM ACTION="http://www.fiu.edu/cgi-bin/gourl" METHOD="GET">
<B><SELECT NAME="URL">
<OPTION VALUE="http://n12.cs.fiu.edu/president/personform.html">Person</OPTION>
```

```
<OPTION VALUE="http://n12.cs.fiu.edu/president/historicalfigureform.html">Historical
Figure</OPTION>
</SELECT> </TD>
<TD><INPUT type="submit"  value="Go">
</TD></TR></font>
</FORM>

<TR><TD><B><FONT size=+1 color="green">Relation Forms: </FONT></B></TD></TR>

<TR><TD ALIGN="right"><font color="black">

<FORM ACTION="http://www.fiu.edu/cgi-bin/gourl" METHOD="GET">
<B><SELECT NAME="URL">
<OPTION VALUE="http://n12.cs.fiu.edu/president/rel_coun_per_form.html">Religion,Country and
Period</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/cont_area_per_form.html">Contribution to Area and
Period</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/coun_popu_form.html">Country and
Population</OPTION>
</SELECT> </TD>
<TD><INPUT type="submit"  value="Go">
</TD></TR></font>
</FORM>

<TR><TD></TD></TR>
<TR><TD align="center" COLSPAN=2><B><FONT SIZE=+1 COLOR="navy">Reports and
Graphs</FONT></B></BR>
<HR SIZE=4 WIDTH="100%"></TD></TR>

<TR><TD><B><FONT size=+1 color="green">General Reports and Graphs: </FONT></B></TD></TR>

<TR><TD ALIGN="right"><font color="black">

<FORM ACTION="http://www.fiu.edu/cgi-bin/gourl" METHOD="GET">
<B><SELECT NAME="URL">
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/repcity.mac/report">City w/Graph</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/repperson.mac/report">Person</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/rephistoricalfigure.mac/report">Historical
Figure</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/repcountry.mac/report">Country</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/repnobelprizefield.mac/report">Nobel Prize
Field</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/sdbwww.exe/repareaofcontribution.mac/report">Area of
Contribution</OPTION>

</SELECT> </TD>
<TD><INPUT type="submit"  value="Go">
</TD></TR></font>
</FORM>

<TR><TD><B><FONT size=+1 color="green">HF Reports and Graphs: </FONT></B></TD></TR>
<TR><TD ALIGN="right"><font color="black">
```

```
<FORM ACTION="http://www.fiu.edu/cgi-bin/gourl" METHOD="GET">
<B><SELECT NAME="URL">
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_city_form.html">HF by City</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_country_form.html">HF by Country
w/Graph</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_country_city_form.html">HF by Country and
City</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_aoc_form.html">HF by Area of Contribution
w/Graph</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_nobelp_form.html">HF by Nobel Prize
w/Graph</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_title_form.html">HF by Title</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_year_form.html">HF by Birth Year</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_span_form.html">HF by Time Span
w/Graph</OPTION>
<OPTION VALUE="http://n12.cs.fiu.edu/president/rephf_country_span_form.html">HF by Country and
Time w/Graph</OPTION>

</SELECT> </TD>
<TD><INPUT type="submit"  value="Go" onClick="return fillForm(this.form)">
</TD></TR></font>
</FORM>

<TR><TD colspan=2><HR SIZE=4 WIDTH="100%"></TD></TR>

<TR><TD align="center" COLSPAN=2><B><FONT SIZE=+1 COLOR="navy">Advanced </br>
Ad-Hoc Retrieving Tools</FONT></B></BR>
<HR SIZE=4 WIDTH="100%"></TD></TR>

<tr><td align="center" colspan=2>
<A
HREF="//n12.cs.fiu.edu/sdbbrowser.exe?DatabaseName=president.DB">
<B> Web-SQL</B></a> </font>
</TD></TR>
<tr><td align="center" colspan=2>
<A
HREF="//n12.cs.fiu.edu/sdbnavigator.exe?DatabaseName=president.DB">
<B> Sem-Access </B></a> </font>
</TD></TR>

</table>

<FORM NAME=TEMP>
<INPUT type=hidden name="NAME_CITY" value="">
<INPUT type=hidden name="NAME_REGION" value="">
<INPUT type=hidden name="NAME_ISPARTOFREGION" value="">
<INPUT type=hidden name="NAME_AREAOFCONTRIBUTION" value="">
<INPUT type=hidden name="NAME_ISPARTOFAOC" value="">
<INPUT type=hidden name="NAME_RELIGION" value="">
<INPUT type=hidden name="NAME_COUNTRY" value="">
<INPUT type=hidden name="SYMBOL" value="">
<INPUT type=hidden name="MAJOR" value="">
<INPUT type=hidden name="SCHOOL_OF" value="">
```

```
<INPUT type=hidden name="PROFESSION" value="">
<INPUT type=hidden name="PERSON_TITLE" value="">
<INPUT type=hidden name="LAST_NAME" value="">
<INPUT type=hidden name="FIRST_NAME" value="">
<INPUT type=hidden name="MIDDLE_NAME" value="">
<INPUT type=hidden name="SUFFIX" value="">
<INPUT type=hidden name="A_K_A" value="">
<INPUT type=hidden name="left_selected" value="">
<INPUT type=hidden name="right_selected" value="">
</FORM>

<FORM NAME=report_form>
<INPUT type=hidden name="title" value="">
<INPUT type=hidden name="right_selected" value="">
<INPUT type=hidden name="list_db" value="">
</FORM>
</center>

<FORM NAME=butt_form>
<INPUT type=hidden name="where_hf" value="">
<INPUT type=hidden name="where_hf1" value="">
<INPUT type=hidden name="where_hf2" value="">
<INPUT type=hidden name="where_hf3" value="">
</FORM>
</center>

<FORM NAME=hf_temp>
<INPUT type=hidden name="born_in" value="">
<INPUT type=hidden name="where_born_in" value="">
<INPUT type=hidden name="worked_in" value="">
<INPUT type=hidden name="where_worked_in" value="">
<INPUT type=hidden name="pupil_of" value="">
<INPUT type=hidden name="has_practiced" value="">
<INPUT type=hidden name="has_won" value="">
<INPUT type=hidden name="has_won_year" value="">
<INPUT type=hidden name="contributed" value="">
<INPUT type=hidden name="contributed_data" value="">
</FORM>

<SCRIPT language=javascript>

        function fillForm(form) {
                document.report_form.title.value="";
                document.report_form.list_db.value="";
                if (form.URL.options[0].selected) {
                        document.report_form.title.value="Cities";
                        document.report_form.list_db.value="CITY";
                        document.report_form.right_selected.value="";
                }
                if (form.URL.options[1].selected) {
                        document.report_form.title.value="Countries";
                        document.report_form.list_db.value="COUNTRY";
                        document.report_form.right_selected.value="";
```

```
                }
                if (form.URL.options[2].selected) {
                        document.report_form.title.value="Countries";
                        document.report_form.list_db.value="COUNTRY";
                        document.report_form.right_selected.value="";
                }
                if (form.URL.options[3].selected) {
                        document.report_form.title.value="Areas of Contribution";
                        document.report_form.list_db.value="AREA_OF_CONTRIBUTION";
                        document.report_form.right_selected.value="";
                }
                if (form.URL.options[4].selected) {
                        document.report_form.title.value="Nobel Prizes";
                        document.report_form.list_db.value="NOBEL_PRIZE_FIELD";
                        document.report_form.right_selected.value="";
                }
                if (form.URL.options[8].selected) {
                        document.report_form.title.value="Countries";
                        document.report_form.list_db.value="COUNTRY";
                        document.report_form.right_selected.value="";
                }
                return true;

        }
</SCRIPT>
</body>
</html>
```

```
%DEFINE{ url = "//n12.cs.fiu.edu/sdbwww.exe/historicalfigurer.mac/report"
%}

%DEFINE{ url_list = "//n12.cs.fiu.edu/sdbwww.exe/hf_list_for_selections.mac/report"
%}

%DEFINE{ url_list_person= "//n12.cs.fiu.edu/sdbwww.exe/hf_list_for_selections_person.mac/report"
%}

%DEFINE{ url_list_npf= "//n12.cs.fiu.edu/sdbwww.exe/hf_year_npf.mac/input"
%}

%DEFINE{ url_list_aoc= "//n12.cs.fiu.edu/sdbwww.exe/hf_data_aoc.mac/input"
%}

%HTML_INPUT{
    <HTML>
    <head>
    <TITLE>Input a new Historical Figure</TITLE>

    </head>
    <body bgcolor="#CCCCFF" text="#000000" link="#0000FF"
     alink="#FF2020" vlink="#800020">
    <H2 ALIGN=CENTER>Input Historical Figure</H2>
    <HR SIZE=4 WIDTH="80%">

        <SCRIPT language=javascript>
                top.frames[0].document.hf_temp.born_in.value="";
                top.frames[0].document.hf_temp.where_born_in.value="";
                top.frames[0].document.hf_temp.worked_in.value="";
                top.frames[0].document.hf_temp.where_worked_in.value="";
                top.frames[0].document.hf_temp.pupil_of.value="";
                top.frames[0].document.hf_temp.has_practiced.value="";
                top.frames[0].document.hf_temp.has_won.value="";
                top.frames[0].document.hf_temp.has_won_year.value="";
                top.frames[0].document.hf_temp.contributed.value="";
                top.frames[0].document.hf_temp.contributed_data.value="";
        </SCRIPT>

    <table border=2>
    <tr><B>PERSON</B><td>
    <FORM NAME="hf_form" target="up"
action="$(url)?what=$(what)&where=$(where)&dob=$(dob)&dod=$(dod)&command_pupil=$(command_
pupil)&command_religion=$(command_religion)" onSubmit="return checkEmpty();">
        <INPUT type=hidden name="what" value="">
        <INPUT type=hidden name="where" value="">
        <INPUT type=hidden name="dob" value="">
        <INPUT type=hidden name="dod" value="">
        <INPUT type=hidden name="parent_hid" value="">
        <INPUT type=hidden name="command_pupil" value="">
```

```
        <INPUT type=hidden name="command_religion" value="">
        <INPUT type=hidden name="command_npf" value="">
        <INPUT type=hidden name="command_aoc" value="">

        <INPUT type=hidden name="db" value="">
        <INPUT type=hidden name="sele" value="">
        <INPUT type=hidden name="born_in_city" value="">
        <INPUT type=hidden name="born_in_country" value="">
        <INPUT type=hidden name="worked_in_city" value="">
        <INPUT type=hidden name="worked_in_country" value="">

    <font size=4 color="#006040" >Last Name: <INPUT type="text" name="last_name" size="22"
value=""></font></br>
    <font size=4 color="#006040" >First Name: <INPUT type="text" name="first_name" size="22"
value=""></font></br>
    <font size=4 color="#006040" >Middle Name: <INPUT type="text" name="middle_name" size="20"
value=""></font></br>
    <font size=4 color="#006040" >Suffix: <INPUT type="text" name="suffix" size="5"
value=""></font></br>
    <font size=4 color="#006040" >Person Title: <INPUT type="text" name="person_title" size="10"
value=""></font></br>
                </br>
    <font size=4 color="#006040" >A.K.A.: <INPUT type="text" name="a_k_a" size="25"
value=""></font>

    </td></tr>
    </table>
<BR></BR>

    <table border=2>
    <tr><B>HISTORICAL FIGURE</B><td>

    <font size=4 color="#006040" >Date of Birth: <INPUT type="text" name="date_of_birth" size="4"
maxlength="4"></font>
    <SELECT NAME = "era_db">
    <OPTION value="AD">AD</OPTION>
    <OPTION value="BC">BC</OPTION>
    </SELECT></br>

    <font size=4 color="#006040" >Date of Death: <INPUT type="text" name="date_of_death" size="4"
maxlength="4"></font>
    <SELECT NAME = "era_dd">
    <OPTION value="AD">AD</OPTION>
    <OPTION value="BC">BC</OPTION>
    </SELECT></br>

    <font size=4 color="#006040" >Birth Order: <INPUT type="text" name="birth_order" size="1"
maxlength="1"></font></BR>
    <br></br>
    <center>
    <A target="right" HREF="$(url_list)?db=CITY&sele='city_born'">was born in City  </a>
    or
```

```
    <A target="right" HREF="$(url_list)?db=COUNTRY&sele='country_born'">was born in
Country</a></br>

    <A target="right" HREF="$(url_list)?db=CITY&sele='city_worked'">worked in City     </a>
    or
    <A target="right" HREF="$(url_list)?db=COUNTRY&sele='country_worked'">worked in
Country</a></br>
   </center>
<br></br>
     <font size=4 color="#006040">Funding: </font>
     <SELECT name="funding">
       <OPTION value="independent">independent</OPTION>
       <OPTION value="patron">patron</OPTION>
       <OPTION value="institution">institution</OPTION>
       <OPTION selected value="other">other</OPTION>
      </SELECT></br>

<font size=4 color="#006040" >No.of References:<INPUT type="text" name="number_of_references"
size="2" ></font></BR>
<font size=4 color="#006040" >Method:<INPUT type="text" name="method" size="2" value=6
maxlength=2></font></BR>
<font size=4 color="#006040" >Parent: <INPUT type="checkbox" name="parent"></font></Br>
<br></br>
<font size=4 color="#006040" >Comments: </font></BR>
 <TEXTAREA name="comments" rows="4" cols="38"></TEXTAREA></font>
 <br></br></br>
<center>
<A target="right" HREF="$(url_list_person)?db=PERSON&sele='pupil_of'">was pupil of
Person</a></BR>
</br></br>
<A target="right" HREF="$(url_list)?db=RELIGION&sele='has_practiced'">has practiced a
Religion</a></BR>
<A target="right" HREF="$(url_list_npf)?db=NOBEL_PRIZE_FIELD&sele='has_won'">has won a Nobel
Prize</a></br>
<A target="right" HREF="$(url_list_aoc)?db=AREA_OF_CONTRIBUTION&sele='contributed'">made a
Contribution</a></br>
</center>
</br></br>
<INPUT TYPE="submit" name="Storing "value="Store the Historical Figure">
</FORM>
</table>
</body>

<SCRIPT language=javascript>

function checkEmpty() {

        var errors="";
        var dob="";
        var dod="";

        if (document.hf_form.last_name.value=="") {
          errors=errors+" -. Error!. You must insert a Last Name.\n" }
```

```
        if (document.hf_form.date_of_birth.value !="") {
                if (document.hf_form.date_of_birth.value < 0) {
                        errors=errors+" -. Error!. You must insert a positive value for Date of Birth.\n";
                } else {
                        if ((document.hf_form.date_of_birth.value > 2050) &&
(document.hf_form.era_db.selectedIndex == 0)) {
                                errors=errors+" -. Error!. You must insert a year before 2050-AD for
Date of Birth.\n";
                        }
                        if ((document.hf_form.date_of_birth.value > 2000)
&&(document.hf_form.era_db.selectedIndex == 1)) {
                                errors=errors+" -. Error!. You must insert a year after 2000-BC for Date
of Birth.\n";
                        }
                }
        }

        if (document.hf_form.date_of_death.value !="") {
                if (document.hf_form.date_of_death.value < 0) {
        errors=errors+" -. Error!. You must insert a positive value for Date of Death.\n";
                } else {
                        if ((document.hf_form.date_of_death.value > 2050) &&
(document.hf_form.era_dd.selectedIndex == 0)) {
                                errors=errors+" -. Error!. You must insert a year before 2050-AD for
Date of Death.\n";
                        }
                        if ((document.hf_form.date_of_death.value > 2000)
&&(document.hf_form.era_dd.selectedIndex == 1)) {
                                errors=errors+" -. Error!. You must insert a year after 2000-BC for Date
of Death.\n";
                        }
                }
        }

        if (document.hf_form.birth_order.value !="") {
                if (document.hf_form.birth_order.value < 0) {
                        errors=errors+" -. Error!. You must insert a positive value for Birth Order.\n";
                }
        }

        if (document.hf_form.number_of_references.value !="") {
                if (document.hf_form.number_of_references.value < 0) {
        errors=errors+" -. Error!. You must insert a positive value for No. of References.\n";
                }
        }

        if (document.hf_form.method.value !="") {
                if (document.hf_form.method.value < 0) {
        errors=errors+" -. Error!. You must insert a positive value for Method.\n";
                }
    }

        if (errors =="") {
```

```
var aux_form_year=0;
var aux_to_year=0;
if (document.hf_form.date_of_birth.value != "" ) {
        if (document.hf_form.era_db.selectedIndex == 1) {

aux_from_year=parseInt((parseInt(document.hf_form.date_of_birth.value,10))*(-1),10);
        }
        if (document.hf_form.era_db.selectedIndex == 0) {
                aux_from_year=parseInt(document.hf_form.date_of_birth.value,10);
        }
}

if (document.hf_form.date_of_death.value != "" ) {
        if (document.hf_form.era_dd.selectedIndex == 1) {

aux_to_year=parseInt((parseInt(document.hf_form.date_of_death.value,10))*(-1),10);
        }
        if (document.hf_form.era_dd.selectedIndex == 0) {
                aux_to_year=parseInt(document.hf_form.date_of_death.value,10);
        }
}

if ( (document.hf_form.date_of_birth.value != "" ) &&
(document.hf_form.date_of_death.value != "" )) {
        if (aux_from_year > aux_to_year) {
                errors=errors+" -. Error!. The date of birth can't come after the date of
death.\n";
        } else {
                document.hf_form.date_of_birth.value=aux_from_year;
                document.hf_form.date_of_death.value=aux_to_year;
        }
} else {
        if (document.hf_form.date_of_birth.value != "" ) {
                document.hf_form.date_of_birth.value=aux_from_year;
        }
        if (document.hf_form.date_of_death.value != "" ) {
                document.hf_form.date_of_death.value=aux_to_year;
        }
}
}


if (errors !="") {
        alert(errors);
        return false;
} else {
        document.hf_form.where.value="LAST_NAME";
        document.hf_form.what.value="'"+document.hf_form.last_name.value+"'";
        document.hf_form.where.value=document.hf_form.where.value+",FIRST_NAME";

document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.first_name.v
alue+"'";
        document.hf_form.where.value=document.hf_form.where.value+",MIDDLE_NAME";
```

```
document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.middle_nam
e.value+"'";
document.hf_form.where.value=document.hf_form.where.value+",SUFFIX";

document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.suffix.value
+"'";
document.hf_form.where.value=document.hf_form.where.value+",PERSON_TITLE";

document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.person_title.
value+"'";
document.hf_form.where.value=document.hf_form.where.value+",A_K_A";

document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.a_k_a.value
+"'";

if (document.hf_form.date_of_birth.value != "") {

document.hf_form.where.value=document.hf_form.where.value+",DATE_OF_BIRTH";

document.hf_form.what.value=document.hf_form.what.value+","+document.hf_form.date_of_birt
h.value;
        if (parseInt(document.hf_form.date_of_birth.value,10) >= 0 ) {
                dob=document.hf_form.date_of_birth.value+"-AD";
        } else {
                dob=(document.hf_form.date_of_birth.value)*(-1)+"-BC";
        }

}
if (document.hf_form.date_of_death.value != "") {

document.hf_form.where.value=document.hf_form.where.value+",DATE_OF_DEATH";

document.hf_form.what.value=document.hf_form.what.value+","+document.hf_form.date_of_deat
h.value;
        if (parseInt(document.hf_form.date_of_death.value,10) >= 0 ) {
                dod=document.hf_form.date_of_death.value+"-AD";
        } else {
                dod=(document.hf_form.date_of_death.value)*(-1)+"-BC";
        }

}

if (document.hf_form.birth_order.value !="") {

document.hf_form.where.value=document.hf_form.where.value+",BIRTH_ORDER";

document.hf_form.what.value=document.hf_form.what.value+","+document.hf_form.birth_order.
value;
}

document.hf_form.where.value=document.hf_form.where.value+",PARENT";
if (document.hf_form.parent.checked) {
```

```
                document.hf_form.what.value=document.hf_form.what.value+",'true'";
                document.hf_form.parent_hid.value="true";
            } else {
                document.hf_form.what.value=document.hf_form.what.value+",'false'";
                document.hf_form.parent_hid.value="false";
            }

            document.hf_form.where.value=document.hf_form.where.value+",FUNDING";

        document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.funding.opti
ons[document.hf_form.funding.selectedIndex].value+"'";

            if (document.hf_form.number_of_references.value != "") {

        document.hf_form.where.value=document.hf_form.where.value+",REFERENCE_NUMBER";

        document.hf_form.what.value=document.hf_form.what.value+","+document.hf_form.number_of_r
eferences.value;
            }
            if (document.hf_form.method.value !="") {
                document.hf_form.where.value=document.hf_form.where.value+",METHOD";

        document.hf_form.what.value=document.hf_form.what.value+","+document.hf_form.method.valu
e;
            }

            if (document.hf_form.comments.value !="") {

        document.hf_form.where.value=document.hf_form.where.value+",COMMENTS";

        document.hf_form.what.value=document.hf_form.what.value+",'"+document.hf_form.comments.v
alue+"'";
            }

            if (top.frames[0].document.hf_temp.born_in.value!="") {

        document.hf_form.where.value=document.hf_form.where.value+","+top.frames[0].document.hf_te
mp.where_born_in.value;

        document.hf_form.what.value=document.hf_form.what.value+",'"+top.frames[0].document.hf_tem
p.born_in.value+"'";
            }

            if (top.frames[0].document.hf_temp.worked_in.value!="") {

        document.hf_form.where.value=document.hf_form.where.value+","+top.frames[0].document.hf_te
mp.where_worked_in.value;

        document.hf_form.what.value=document.hf_form.what.value+",'"+top.frames[0].document.hf_tem
p.worked_in.value+"'";
            }
```

```
what=document.hf_form.what.value;

where=document.hf_form.where.value;

document.hf_form.dob.value=dob;
document.hf_form.dod.value=dod;

if (top.frames[0].document.hf_temp.pupil_of.value!="") {
        var one = "insert into HISTORICAL_FIGURE__PUPIL_OF
(HISTORICAL_FIGURE, PERSON) ";
        var two = "select HISTORICAL_FIGURE, PERSON  from
HISTORICAL_FIGURE, PERSON ";
        var three_one= "where ( (
HISTORICAL_FIGURE.LAST_NAME='"+document.hf_form.last_name.value+"' and
HISTORICAL_FIGURE.FIRST_NAME='"+document.hf_form.first_name.value+"' ";
        var three_two= "and
HISTORICAL_FIGURE.MIDDLE_NAME='"+document.hf_form.middle_name.value+"' and
HISTORICAL_FIGURE.SUFFIX='"+document.hf_form.suffix.value+"' ";
        var three_three= "and
HISTORICAL_FIGURE.PERSON_TITLE='"+document.hf_form.person_title.value+"' and
HISTORICAL_FIGURE.A_K_A='"+document.hf_form.a_k_a.value+"' ) ";
        var three=three_one+three_two+three_three;

        var myArray= new Array();
        var myString=top.frames[0].document.hf_temp.pupil_of.value;
        var myArray= myString.split(",");
        var itemCount = myArray.length-2;
        var myPerson=myArray[0];
        var myTokens= myPerson.split("*");

        var four_cero= "and ( ( PERSON.LAST_NAME='"+myTokens[3]+"' and
PERSON.FIRST_NAME='"+myTokens[1]+"' and PERSON.MIDDLE_NAME='"+myTokens[2]+"' ";
        var four_one= "and PERSON.SUFFIX='"+myTokens[4]+"' and
PERSON.PERSON_TITLE='"+myTokens[0]+"' and PERSON.A_K_A='"+myTokens[5]+"' ) ";
        var four=four_cero+four_one;

        for (var i=1;i<=itemCount; i++) {
                var myPerson=myArray[i];
                var myTokens= myPerson.split("*");
                var four_cero= "or ( PERSON.LAST_NAME='"+myTokens[3]+"' and
PERSON.FIRST_NAME='"+myTokens[1]+"' and PERSON.MIDDLE_NAME='"+myTokens[2]+"' ";
                var four_one= "and PERSON.SUFFIX='"+myTokens[4]+"' and
PERSON.PERSON_TITLE='"+myTokens[0]+"' and PERSON.A_K_A='"+myTokens[5]+"' ) ";
                four=four+four_cero+four_one;
        }

        four = four+") )";

        document.hf_form.command_pupil.value=one+two+three+four;

        command_pupil=document.hf_form.command_pupil.value;

    } else {
```

```
                        document.hf_form.command_pupil.value="select * from PERSON where
LAST_NAME=' ";
                }

                if (top.frames[0].document.hf_temp.has_practiced.value!="") {
                        var one = "insert into HISTORICAL_FIGURE__HAS_PRACTICED
(HISTORICAL_FIGURE, RELIGION) ";
                        var two = "select HISTORICAL_FIGURE, RELIGION  from
HISTORICAL_FIGURE, RELIGION ";
                        var three_one= "where ( (
HISTORICAL_FIGURE.LAST_NAME='"+document.hf_form.last_name.value+"' and
HISTORICAL_FIGURE.FIRST_NAME='"+document.hf_form.first_name.value+"' ";
                        var three_two= "and
HISTORICAL_FIGURE.MIDDLE_NAME='"+document.hf_form.middle_name.value+"' and
HISTORICAL_FIGURE.SUFFIX='"+document.hf_form.suffix.value+"' ";
                        var three_three= "and
HISTORICAL_FIGURE.PERSON_TITLE='"+document.hf_form.person_title.value+"' and
HISTORICAL_FIGURE.A_K_A='"+document.hf_form.a_k_a.value+"' ) ";
                        var three=three_one+three_two+three_three;

                        var myArray= new Array();
                        var myString=top.frames[0].document.hf_temp.has_practiced.value;
                        var myArray= myString.split(",");
                        var itemCount = myArray.length-2;

                        var myReligion=myArray[0];

                        var four= "and ( RELIGION.NAME='"+myArray[0]+"'";

                        for (var i=1;i<=itemCount; i++) {
                                var myPerson=myArray[i];
                                four=four+" or RELIGION.NAME='"+myArray[i]+"'";
                        }

                        four = four+") )";

                        document.hf_form.command_religion.value=one+two+three+four;

                        command_religion=document.hf_form.command_religion.value;


                } else {
                        document.hf_form.command_religion.value="select * from PERSON where
LAST_NAME=' ";
                }

                if (top.frames[0].document.hf_temp.has_won.value!="") {
                        var one = "insert into HISTORICAL_FIGURE___WON_BY
(HISTORICAL_FIGURE, NOBEL_PRIZE) ";
                        var two = "select distinct HISTORICAL_FIGURE, NOBEL_PRIZE  from
HISTORICAL_FIGURE, NOBEL_PRIZE ";
```

```
                    var three_one= "where ( (
HISTORICAL_FIGURE.LAST_NAME='"+document.hf_form.last_name.value+"' and
HISTORICAL_FIGURE.FIRST_NAME='"+document.hf_form.first_name.value+"' ";
                    var three_two= "and
HISTORICAL_FIGURE.MIDDLE_NAME='"+document.hf_form.middle_name.value+"' and
HISTORICAL_FIGURE.SUFFIX='"+document.hf_form.suffix.value+"' ";
                    var three_three= "and
HISTORICAL_FIGURE.PERSON_TITLE='"+document.hf_form.person_title.value+"' and
HISTORICAL_FIGURE.A_K_A='"+document.hf_form.a_k_a.value+"' ) ";
                    var three=three_one+three_two+three_three;

                    var myArray_Prize= new Array();
                    var myArray_Year= new Array();

                    var myString_Prize=top.frames[0].document.hf_temp.has_won.value;
                    var myString_Year=top.frames[0].document.hf_temp.has_won_year.value;

                    var myArray_Prize= myString_Prize.split(",");
                    var myArray_Year= myString_Year.split(",");
                    var itemCount = myArray_Prize.length-2;

                    var four= "and ( (NOBEL_PRIZE.YEAR="+myArray_Year[0]+" and
NOBEL_PRIZE.IN__NAME='"+myArray_Prize[0]+"') ";

                    for (var i=1;i<=itemCount; i++) {
                            four=four+" or (NOBEL_PRIZE.YEAR="+myArray_Year[i]+" and
NOBEL_PRIZE.IN__NAME='"+myArray_Prize[i]+"') ";
                    }

                    four = four+") )";

                    document.hf_form.command_npf.value=one+two+three+four;

                    command_npf=document.hf_form.command_npf.value;

            } else {
                    document.hf_form.command_npf.value="select * from PERSON where
LAST_NAME='' ";
            }


            if (top.frames[0].document.hf_temp.contributed.value!="") {
                    var one = "insert into HISTORICAL_FIGURE___BY
(HISTORICAL_FIGURE, CONTRIBUTION_TO_AREA) ";
                    var two = "select distinct HISTORICAL_FIGURE,
CONTRIBUTION_TO_AREA from HISTORICAL_FIGURE, CONTRIBUTION_TO_AREA ";
                    var three_one= "where ( (
HISTORICAL_FIGURE.LAST_NAME='"+document.hf_form.last_name.value+"' and
HISTORICAL_FIGURE.FIRST_NAME='"+document.hf_form.first_name.value+"' ";
                    var three_two= "and
HISTORICAL_FIGURE.MIDDLE_NAME='"+document.hf_form.middle_name.value+"' and
HISTORICAL_FIGURE.SUFFIX='"+document.hf_form.suffix.value+"' ";
```

```
                        var three_three= "and
HISTORICAL_FIGURE.PERSON_TITLE="'+document.hf_form.person_title.value+"' and
HISTORICAL_FIGURE.A_K_A="'+document.hf_form.a_k_a.value+"' ) ";
                        var three=three_one+three_two+three_three;

                        var myArray_Area= new Array();
                        var myArray_Data= new Array();

                        var myString_Area=top.frames[0].document.hf_temp.contributed.value;
                        var myString_Data=top.frames[0].document.hf_temp.contributed_data.value;

                        var myArray_Area= myString_Area.split(",");
                        var myArray_Data= myString_Data.split(",");
                        var itemCount = myArray_Area.length-2;

                        var myCell_Data= myArray_Data[0].split("*");

                        var four= "and (
(CONTRIBUTION_TO_AREA.MAJOR="'+myCell_Data[0]+"' and
CONTRIBUTION_TO_AREA.SCHOOL_OF="'+myCell_Data[1]+"' and
CONTRIBUTION_TO_AREA.PROFESSION="'+myCell_Data[2]+"' and
CONTRIBUTION_TO_AREA.OF__NAME="'+myArray_Area[0]+"') ";

                        for (var i=1;i<=itemCount; i++) {
                                var myCell_Data= myArray_Data[i].split("*");
                                four=four+" or
(CONTRIBUTION_TO_AREA.MAJOR="'+myCell_Data[0]+"' and
CONTRIBUTION_TO_AREA.SCHOOL_OF="'+myCell_Data[1]+"' and
CONTRIBUTION_TO_AREA.PROFESSION="'+myCell_Data[2]+"' and
CONTRIBUTION_TO_AREA.OF__NAME="'+myArray_Area[i]+"') ";
                        }

                        four = four+") )";

                        document.hf_form.command_aoc.value=one+two+three+four;

                        command_aoc=document.hf_form.command_aoc.value;

                } else {
                        document.hf_form.command_aoc.value="select * from PERSON where
LAST_NAME=" ";
                }


                return true;
    }


  }

        </SCRIPT>
      </html>
  %}
```

```
%DEFINE DATABASE = "president.DB"

%SDB(sdb01) {
        INSERT INTO HISTORICAL_FIGURE ($(where)) VALUES ($(what))

%SDB_REPORT{
%ROW{



%}
%}
%}

%SDB(sdb02) {
        $(command_pupil)

%SDB_REPORT{
%ROW{
.


%}
%}
%}

%SDB(sdb03) {
        $(command_religion)

%SDB_REPORT{
%ROW{
.


%}
%}
%}

%SDB(sdb04) {
        $(command_npf)

%SDB_REPORT{
%ROW{
.


%}
%}
%}

%SDB(sdb05) {
```

```
        $(command_aoc)

%SDB_REPORT{
%ROW{
.


%}
%}
%}

%HTML_REPORT{
     <HTML>
     <head>

     <TITLE>Input a new Historical Figure</TITLE>
     </head>
     <body bgcolor="#CCCCFF" text="#000000" link="#0000FF"
      alink="#FF2020" vlink="#800020">
                    %EXEC_SQL(sdb01)
                    %EXEC_SQL(sdb02)
                    %EXEC_SQL(sdb03)
                    %EXEC_SQL(sdb04)
                    %EXEC_SQL(sdb05)

<center>
<TABLE border="2" width="420">
<TR>
<TD align="center"> <font color="navy"><B>Inserted the following Historical Figure into the database
...</B></font></TD>
</TR>
<TR>
<TD> Last Name:          <font color="#006666">$(last_name)</font></br>
First Name:       <font color="#006666">$(first_name) </font></br>
Middle Name:      <font color="#006666">$(middle_name)</font></br>
Suffix: <font color="#006666">$(suffix) </font></br>
Person Title: <font color="#006666">$(person_title) </font></br>
A_K_A: <font color="#006666">$(a_k_a) </font></br>
</br>
Date of Birth: <font color="#006666">$(dob) </font></br>
Date of Death: <font color="#006666">$(dod) </font></br>
Birth order: <font color="#006666">$(birth_order) </font></br>
</br>
was born in:
<font color="#006666">
<SCRIPT language=javascript>
document.write(top.frames[0].document.hf_temp.born_in.value);
</SCRIPT>
</font></br>
worked in:
<font color="#006666">
<SCRIPT language=javascript>
document.write(top.frames[0].document.hf_temp.worked_in.value);
```

```
</SCRIPT>
</font></br>
</br>
Funding: <font color="#006666">$(funding) </font></br>
No. of References: <font color="#006666">$(number_of_references) </font></br>
Method: <font color="#006666">$(method) </font></br>
Parent: <font color="#006666">$(parent_hid) </font></br>
Comments: <font color="#006666">$(comments) </font></br>
</br>
was pupil of:
<font color="#006666">
<SCRIPT language=javascript>
var myArray= new Array();
var myString=top.frames[0].document.hf_temp.pupil_of.value;
var myArray= myString.split(",");
var itemCount = myArray.length-2;
for (var i=0;i<=itemCount; i++) {
        var myPerson=myArray[i];
        var myTokens= myPerson.split("*");
        for ( var j=0; j<5; j++) {
                document.write(myTokens[j]+" ");
        }
        if (myTokens[5] != "") {
                document.write("("+myTokens[5]+")");
        }
        document.write(",");
}
</SCRIPT>
</font></br></br>
has practiced a Religion:
<font color="#006666">
<SCRIPT language=javascript>
var myArray= new Array();
var myString=top.frames[0].document.hf_temp.has_practiced.value;
var myArray= myString.split(",");
var itemCount = myArray.length-2;
for (var i=0;i<=itemCount; i++) {
        document.write(myArray[i]+",");
}
</SCRIPT>
</font></br>
</br>
has won a Nobel Prize:
<font color="#006666">
<SCRIPT language=javascript>
var myArray_Prize= new Array();
var myArray_Year=new Array();
var myString_Prize=top.frames[0].document.hf_temp.has_won.value;
var myString_Year=top.frames[0].document.hf_temp.has_won_year.value;
var myArray_Prize= myString_Prize.split(",");
var myArray_Year= myString_Year.split(",");
var itemCount = myArray_Prize.length-2;
for (var i=0;i<=itemCount; i++) {
```

```
        document.write(myArray_Prize[i]+"("+myArray_Year[i]+"),");
}
</SCRIPT>
</font></br>
</br>
made a Contribution in:
<font color="#006666">
<SCRIPT language=javascript>
var myArray_Area= new Array();
var myArray_Data= new Array();
var myString_Area=top.frames[0].document.hf_temp.contributed.value;
var myString_Data=top.frames[0].document.hf_temp.contributed_data.value;
var myArray_Area= myString_Area.split(",");
var myArray_Data= myString_Data.split(",");
var itemCount = myArray_Area.length-2;
for (var i=0;i<=itemCount; i++) {
        var myCell_Data= myArray_Data[i].split("*");
        document.write(myArray_Area[i]+"(major: "+myCell_Data[0]+", school of: "+myCell_Data[1]+",
profession: "+myCell_Data[2]+"), ");
}
</SCRIPT>
</br>
</TD>
</TR>
</table>
</center>
<P><A target="operations" HREF="http://n12.cs.fiu.edu/president/historicalfigureform.html">Reload List
of Historical Figures and Insert a new one</A></P>


</body>
</html>
%}
```

```
%DEFINE DATABASE = "president.DB"

%DEFINE{ url = "//n12.cs.fiu.edu/sdbwww.exe/modhistoricalfigure.mac/report"
%}

%SDB(sdb01) {
        SELECT
LAST_NAME,FIRST_NAME,MIDDLE_NAME,SUFFIX,PERSON_TITLE,A_K_A,DATE_OF_BIRTH,
DATE_OF_DEATH,BORN_IN__NAME,BORN_IN_COUNTRY__NAME,WORKED_IN__NAME,WO
RKED_IN_COUNTRY__NAME,BIRTH_ORDER,FUNDING,REFERENCE_NUMBER,METHOD,COM
MENTS FROM HISTORICAL_FIGURE

%SDB_REPORT{
 <center>
 <TABLE border="2" width="420">
 <TH>Historical Figure</TH> <TH>A.K.A.</TH>
        %ROW{
                <TR>
                <TD> <A target="left"
HREF="$(url)?LAST_NAME=$(V0)&FIRST_NAME=$(V1)&MIDDLE_NAME=$(V2)&SUFFIX=$(V3
)&PERSON_TITLE=$(V4)&A_K_A=$(V5)&DATE_OF_BIRTH=$(V6)&DATE_OF_DEATH=$(V7)&
BIRTH_ORDER=$(V12)&FUNDING=$(V13)&REFERENCE_NUMBER=$(V14)&METHOD=$(V15)&
COMMENTS=$(V16)&BORN_IN__NAME=$(V8)&BORN_IN_COUNTRY__NAME=$(V9)&WORKE
D_IN__NAME=$(V10)&WORKED_IN_COUNTRY__NAME=$(V11)">$(V4) $(V1) $(V2) $(V0)
$(V3)</a></TD>
               <TD> $(V5) </TD>
                </TR>
                %}
</TABLE>
</center>
%}
%}

%HTML_REPORT{
        <html>
        <head>
        <BASE target="down">
        <TITLE>List of Historical Figures to Modify/Delete</TITLE>
        </head>
        <body bgcolor="#CCCCFF" text="#0000000" link="#0000FF"
                alink="#FF2020" vlink="#800020">
        <center>
        <H2 ALIGN=CENTER> Modify/Remove Historical Figure</H2>
     <HR SIZE=4 WIDTH="80%">
        <font color="navy"><B>Select a particular Historical Figure to Modify/Remove...</B></font>
        <br></br>
        </center>
        %EXEC_SQL(sdb01)
        </body>
        </html>
 %}
```

```
%DEFINE{ url1 = "//n12.cs.fiu.edu/sdbwww.exe/rephf_display_country_span.mac/report"
%}

%HTML_INPUT{
    <HTML>
    <head>
    <TITLE>Report</TITLE>
    </head>
    <body bgcolor="#CCCCFF" text="#000000" link="#0000FF" alink="#FF2020" vlink="#800020">
    <H2 ALIGN=CENTER>Historical Figure Report by Time Span and Country</H2>

    <HR SIZE=4 WIDTH="100%">
    </center>
    <table border=0>
    <tr><td>

  <FORM NAME = "report_selection1_form"  target="operations"
action="$(url1)?command=$(command)&first=$(first)&rest=$(rest)">
  <INPUT type=hidden name="command" value="">
  <INPUT type=hidden name="first" value="">
  <INPUT type=hidden name="rest" value="">

  <font size=4 color="#006040" >from Year: <INPUT type="text" name="from_year" size="4"
maxlength="4"></font>
  <SELECT NAME = "era_fy">
  <OPTION value="AD">AD</OPTION>
  <OPTION value="BC">BC</OPTION>
  </SELECT></br>

  <font size=4 color="#006040" >to Year: <INPUT type="text" name="to_year" size="4"
maxlength="4"></font>
  <SELECT NAME = "era_ty">
  <OPTION value="AD">AD</OPTION>
  <OPTION value="BC">BC</OPTION>
  </SELECT></br>

  <font size=4 color="#006040" >Generate the report by Country(ies) for the inserted Period ...</font>


  <INPUT TYPE="submit" name="display" value="Go" onClick="return createParam1(this.form)">


  </td></tr>

  </form>
  </table>
```

```
<SCRIPT language=javascript>

function createParam1(form) {

        var errors="";

 if (document.report_selection1_form.from_year.value=="") {
        errors=errors+" -. Error!. You must insert the year to start.\n";
 }
 if (document.report_selection1_form.to_year.value=="") {
        errors=errors+" -. Error!. You must insert the year to end\n";
 }

 if (document.report_selection1_form.from_year.value !="") {
  if (document.report_selection1_form.from_year.value < 0) {
      errors=errors+" -. Error!. You must insert a positive value for the year to start.\n";
  } else {
     if ((document.report_selection1_form.from_year.value > 2050) &&
(document.report_selection1_form.era_fy.selectedIndex == 0)) {
      errors=errors+" -. Error!. You must insert a year before 2050-AD.\n";
     }
     if ((document.report_selection1_form.from_year.value > 2000)
&&(document.report_selection1_form.era_fy.selectedIndex == 1)) {
            errors=errors+" -. Error!. You must insert a year after 2000-BC.\n";
     }
  }
 }

 if (document.report_selection1_form.to_year.value !="") {
  if (document.report_selection1_form.to_year.value < 0) {
      errors=errors+" -. Error!. You must insert a positive value for the year to end.\n";
  } else {
     if ((document.report_selection1_form.to_year.value > 2050) &&
(document.report_selection1_form.era_ty.selectedIndex == 0)) {
      errors=errors+" -. Error!. You must insert a year before 2050-AD.\n";
     }
     if ((document.report_selection1_form.to_year.value > 2000)
&&(document.report_selection1_form.era_ty.selectedIndex == 1)) {
      errors=errors+" -. Error!. You must insert a year after 2000-BC.\n";
     }
  }
 }


 if (errors =="") {
        var aux_form_year=0;
        var aux_to_year=0;
        if (document.report_selection1_form.from_year.value != "" ) {
                if (document.report_selection1_form.era_fy.selectedIndex == 1) {

        aux_from_year=parseInt((parseInt(document.report_selection1_form.from_year.value,10))*(-
1),10);
                }
```

```
                if (document.report_selection1_form.era_fy.selectedIndex == 0) {

        aux_from_year=parseInt(document.report_selection1_form.from_year.value,10);
                }
        }

                if (document.report_selection1_form.to_year.value != "" ) {
                        if (document.report_selection1_form.era_ty.selectedIndex == 1) {

        aux_to_year=parseInt((parseInt(document.report_selection1_form.to_year.value,10))*(-1),10);
                        }
                        if (document.report_selection1_form.era_ty.selectedIndex == 0) {

        aux_to_year=parseInt(document.report_selection1_form.to_year.value,10);
                        }
                }

                if ( (document.report_selection1_form.from_year.value != "" ) &&
(document.report_selection1_form.to_year.value != "" )) {
                        if (aux_from_year > aux_to_year) {
                                errors=errors+" -. Error!. The year for the Period to start can't come
after the year for the Period to end.\n";
                        } else {
                                document.report_selection1_form.from_year.value=aux_from_year;
                                document.report_selection1_form.to_year.value=aux_to_year;
                        }
                } else {
                        if (document.report_selection1_form.from_year.value != "" ) {
                                document.report_selection1_form.from_year.value=aux_from_year;
                        }
                        if (document.report_selection1_form.to_year.value != "" ) {
                                document.report_selection1_form.to_year.value=aux_to_year;
                        }
                }
        }

        if (top.frames[0].document.report_form.right_selected.value == "" ) {
            errors=errors+" -. Error!. You must select at least one element from the list";
        }


        if (errors !="") {
          alert(errors);
          return false;
        } else {
                document.report_selection1_form.command.value="select
LAST_NAME,FIRST_NAME,MIDDLE_NAME,PERSON_TITLE,SUFFIX,A_K_A,DATE_OF_BIRTH,
DATE_OF_DEATH,BORN_IN_COUNTRY__NAME,BORN_IN__BELONGS_TO__NAME from
HISTORICAL_FIGURE";

        var token =
top.frames[0].document.report_form.right_selected.value.substring(0,top.frames[0].document.report_form.r
ight_selected.value.indexOf(","));
```

```
        top.frames[0].document.report_form.right_selected.value=top.frames[0].document.report_form.rig
ht_selected.value.substring(token.length+1,top.frames[0].document.report_form.right_selected.value.lengt);
        var where_value_coun="( ( BORN_IN_COUNTRY__NAME="'+token+"' or
BORN_IN__BELONGS_TO__NAME="'+token+"'";
        document.report_selection1_form.first.value=token;
                while (top.frames[0].document.report_form.right_selected.value.length > 0) {
                        var token =
top.frames[0].document.report_form.right_selected.value.substring(0,top.frames[0].document.report_form.r
ight_selected.value.indexOf(","));

        top.frames[0].document.report_form.right_selected.value=top.frames[0].document.report_form.rig
ht_selected.value.substring(token.length+1,top.frames[0].document.report_form.right_selected.value.lengt);
                        where_value_coun=where_value_coun+" or
BORN_IN_COUNTRY__NAME="'+token+"' or BORN_IN__BELONGS_TO__NAME="'+token+"'";
                        document.report_selection1_form.rest.value=
        document.report_selection1_form.rest.value+token+","
                }
                where_value_coun=where_value_coun+" ) ";
        document.report_selection1_form.command.value=document.report_selection1_form.command.va
lue+" where "+where_value_coun;
                var where_value1=" DATE_OF_BIRTH >=
"+document.report_selection1_form.from_year.value+" and DATE_OF_BIRTH <=
"+document.report_selection1_form.to_year.value+" and DATE_OF_DEATH >=
"+document.report_selection1_form.from_year.value+" and DATE_OF_DEATH <=
"+document.report_selection1_form.to_year.value;
                var where_value2=" DATE_OF_BIRTH <
"+document.report_selection1_form.from_year.value+" and DATE_OF_BIRTH <
"+document.report_selection1_form.to_year.value+" and DATE_OF_DEATH >=
"+document.report_selection1_form.from_year.value+" and DATE_OF_DEATH <=
"+document.report_selection1_form.to_year.value;
                var where_value3=" DATE_OF_BIRTH <
"+document.report_selection1_form.from_year.value+" and DATE_OF_BIRTH <
"+document.report_selection1_form.to_year.value+" and DATE_OF_DEATH >
"+document.report_selection1_form.from_year.value+" and DATE_OF_DEATH >
"+document.report_selection1_form.to_year.value;
                var where_value4=" DATE_OF_BIRTH >=
"+document.report_selection1_form.from_year.value+" and DATE_OF_BIRTH <
"+document.report_selection1_form.to_year.value+" and DATE_OF_DEATH >
"+document.report_selection1_form.from_year.value+" and DATE_OF_DEATH >
"+document.report_selection1_form.to_year.value;
                var where_value=" and  ( ("+where_value1+") or ("+where_value2+") or
("+where_value3+") or ("+where_value4+") ) )";
        document.report_selection1_form.command.value=document.report_selection1_form.command.va
lue+where_value+" order by BORN_IN__BELONGS_TO__NAME,DATE_OF_BIRTH";
                command=document.report_selection1_form.command.value;
                return true;
        }
}
</SCRIPT>
</body>
</html>
%}
```

```
%DEFINE DATABASE = "president.DB"

%SDB(sdb01) {
        $(command)

%SDB_REPORT{
    <center>
                <TABLE border=2 width="400">
                <th>Country</TH> <th>Date of Birth</TH> <th>Date of Death</TH> <TH>Historical
Figure</TH> <TH>A_K_A</TH>
                %ROW{
                        <TR>
                        <td>$(V9)</td>
                        <TD align="right">
                        <SCRIPT language=javascript>
                        if ('$(V6)' != '(null)') {
                                if ( $(V6) >= 0  ) {
                                        document.write($(V6)+"-AD");
                                } else {
                                        var posi_year=$(V6)*(-1);
                                        document.write(posi_year+"-BC");
                                }
                        }
                        </SCRIPT>
                        </TD>
                        <TD align="right">
                        <SCRIPT language=javascript>
                        if ('$(V7)' != '(null)') {
                                if ( $(V7) >= 0  ) {
                                        document.write($(V7)+"-AD");
                                } else {
                                        var posi_year=$(V7)*(-1);
                                        document.write(posi_year+"-BC");
                                }
                        }
                        </SCRIPT>
                        </TD>
                        <TD> $(V3) $(V1) $(V2) $(V0) $(V4)</TD>
        <TD> $(V5) </TD>
        </TR>

                                <SCRIPT language=javascript>
                                if ($(V6) < $(V7) ) {
                                var j=$(V6);
                                do {
                                        if ("$(V9)" != "$(first)") {
                                                temp=myArray[j];
                                                temp=temp+1;
                                                myArray[j]=temp;
```

```
                                        }

                                        if ("$(V9)" == "$(first)") {
                                                temp=myArray2[j];
                                                temp=temp+1;
                                                myArray2[j]=temp;
                                        }

                                        j=j+1;
                                } while (j <= $(V7))
                                }

                        </SCRIPT>
                %}
                </table>
                </center>

        %}
%}



%HTML_REPORT{
        <html>
        <head>
  <TITLE>Report</TITLE>
        </head>
                <SCRIPT language=javascript>

        var gridx="0";
        var gridy="0";
        var grid2x="0";
        var grid2y="0";
        var max_y=0;
        var sep_x="0";

   function Constr_X(gridx) {

                for ( var i=-2000; i <= total_yr; i++) {
                        if ( (myArray[i] > 0) && (i != 0) )  {
                                gridx=gridx+","+i;
                        }
                }
                tempo_gridx=gridx.substr(2,gridx.length);
                gridx=tempo_gridx;
                // alert(gridx);
                return gridx;
          }

          function Constr2_X(grid2x) {

                for ( var i=-2000; i <= total_yr; i++) {
                        if ( (myArray2[i] > 0) && (i != 0) )  {
```

```
                        grid2x=grid2x+","+i;
                }
        }
        tempo_grid2x=grid2x.substr(2,grid2x.length);
        grid2x=tempo_grid2x;
//      alert(grid2x);
        return grid2x;
}

function Constr_Y(gridy) {

        for ( var i=-2000; i <= total_yr; i++) {
                if ( (myArray[i] > 0) && (i != 0) ) {
                        gridy=gridy+","+myArray[i];
                }
        }
        tempo_gridy=gridy.substr(2,gridy.length);
        gridy=tempo_gridy;
        // alert(gridy);
        return gridy;
}

function Constr2_Y(grid2y) {

        for ( var i=-2000; i <= total_yr; i++) {
                if ( (myArray2[i] > 0) && (i != 0) ) {
                        grid2y=grid2y+","+myArray2[i];
                }
        }
        tempo_grid2y=grid2y.substr(2,grid2y.length);
        grid2y=tempo_grid2y;
        // alert(grid2y);
        return grid2y;
}

function maxY(max_y) {

        for ( var i=-2000; i <= total_yr; i++) {
                if  (myArray[i] > max_y) {
                        max_y=myArray[i];
                }
        }
        max_y=(Math.round(max_y/5)+1)*5;
        // alert(max_y);
        return max_y;
}


    </SCRIPT>
<body bgcolor="#CCCCFF" text="#000000" link="#0000FF"
 alink="#FF2020" vlink="#800020">
<H2 ALIGN=CENTER>Historical Figures by Country and Time Span</H2>
    <H3 ALIGN=CENTER>(including the Years inserted)</H3>
```

```
Calculating ...
<SCRIPT language=javascript>
   var total_yr=4051;

        var myArray= new Array(total_yr);
        var myArray2= new Array(total_yr);

        for ( var i=-2000; i <= total_yr; i++) {
                myArray[i]=0;
                myArray2[i]=0;
        }

</SCRIPT>
%EXEC_SQL(sdb01)
</br>
</br>

<applet code=javachart.applet.lineApp.class  codebase="http://n12.cs.fiu.edu/president" width=500
height=400>
        <param name=titleString value="Distribution of Historical Figures during time">
        <param name=titleFont value="TimesRoman,18,1">
        <param name=xAxisTitle value="Years">
        <param name=xAxisTitleColor value="blue">
        <param name=yAxisTitle value="No.of HFs">
        <param name=yAxisTitleColor value="blue">
        <param name=legendOn value="true">
        <param name=xAxisOptions value="gridOn,noAutoScale">
        <param name=xAxisGridCount value="1">
        <param name=xAxisStart value=$(from_year)>
        <param name=xAxisEnd value=$(to_year)>
        <param name=yAxisOptions value="gridOn,AutoScale">
        <param name=yAxisStart value="0">
        <param name=dataset0yValues value=&{Constr_Y(gridy);};>
        <param name=dataset0xValues value=&{Constr_X(gridx);};>
        <param name=dataset1yValues value=&{Constr2_Y(grid2y);};>
        <param name=dataset1xValues value=&{Constr2_X(grid2x);};>
        <param name=dataset0Name value="$(rest)">
        <param name=dataset1Name value="$(first)">
        <param name=legendllX value=0.01>
        <param name=legendllY value=0.01>
        <param name=iconWidth value=0.1>
        <param name=iconHeight value=0.02>
        <param name=yAxisEnd value=&{maxY(max_y);};>
</applet>
</body>
</html>
%}
```