1992

Towards the Integration of Object-Oriented Constructs within Structured Query Language (SQL)

Paul Francis Rabuck

University of North Florida

Suggested Citation
Rabuck, Paul Francis, "Towards the Integration of Object-Oriented Constructs within Structured Query Language (SQL)" (1992). UNF Graduate Theses and Dissertations. 226.
https://digitalcommons.unf.edu/etd/226
TOWARDS THE INTEGRATION OF OBJECT-ORIENTED CONSTRUCTS WITHIN STRUCTURED QUERY LANGUAGE (SQL)

by

Paul Francis Rabuck

A thesis submitted to the College of Computing Sciences and Engineering in partial fulfillment of the requirements for the degree of

Master of Science in Computer and Information Sciences

UNIVERSITY OF NORTH FLORIDA
COLLEGE OF COMPUTING SCIENCES AND ENGINEERING

December, 1992
Copyright © 1992 by Paul Francis Rabuck

All rights reserved. Reproduction in whole or in part in any form requires the prior written permission of Paul Francis Rabuck or designated representative.
The thesis "Towards the Integration of Object-oriented Constructs Within the Structured Query Language (SQL)" submitted by Paul Francis Rabuck in partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences has been approved by the thesis committee:

Approved by the thesis committee: Date

Signature Deleted 11/9/1992
Dr. Susan R. Wallace
Thesis Advisor and Committee Chairperson

Signature Deleted 11/9/92
Dr. Judith Solano

Signature Deleted 11/9/92
Dr. F. Loyne Wallace

Accepted for the College of Computing Sciences and Engineering:

Signature Deleted
Dr. Robert F. Roggio
Dean of the College

Accepted for the University:

Signature Deleted 11/9/92
Dr. Charles Galloway
Dean of Graduate Studies
ACKNOWLEDGMENT

Special thanks go to my parents, Leo V. and Cecelia Rabuck, for their love and continuing support in this and all endeavors that I have pursued; and to the thesis committee, for their many hours of work on my behalf.
CONTENTS

List of Figures ......................................................... vii
Abstract ................................................................. viii
Chapter 1: Introduction .............................................. 1
Chapter 2: The Project .................................................. 5
  2.1 Selecting a semantic data model ............... 5
  2.2 Selecting tools for development ............... 8
  2.3 Understanding OSAM* ..................... 10
Chapter 3: Designing the User Interface ............. 15
  3.1 OSAM* Concerns .............................................. 15
  3.2 SQL Concerns .................................................. 18
Chapter 4: Implementing the OSAM* Designer ........ 20
  4.1 The Workspace: Showing the s-diagram ... 20
     4.1.1 Representing OSAM* objects .......... 22
  4.2 Accommodating SQL in the prototype ....... 35
     4.2.1 Representing OSAM* attributes .... 36
  4.3 Considerations for Microsoft Windows ...... 47
  4.4 Generating SQL from OSAM* .................. 52
  4.5 Results of the Project ....................... 56
Chapter 5: Future Areas of Study ....................... 59
  5.1 Evaluation of Design Techniques ............. 59
  5.2 Improving the Graphical User Interface .... 60
  5.3 Possible Enhancements to SQL ................. 61
  5.4 Other Issues ................................................. 63
References ............................................................... 64
CONTENTS

Appendix A: OSAM* Designer Code Listing ............ 66
Vita ............................................................ 200
FIGURES

Figure 1: A Simple S-diagram ......................... 6
Figure 2: Main Window ................................. 21
Figure 3: Object Menu ................................. 22
Figure 4: Create An Object Dialog .................... 23
Figure 5: Associate An Object Dialog .................. 27
Figure 6: Disassociate An Object Dialog ............... 28
Figure 7: Find An Object Dialog ....................... 30
Figure 8: Workspace "Cells" ............................ 32
Figure 9: Object Attributes Dialog .................... 41
Figure 10: Summary Attributes Dialog .................. 44
Figure 11: Object Rules Dialog ......................... 46
Figure 12: File Menu ................................. 47
Figure 13: Open File Common Dialog ..................... 48
Figure 14: Edit Menu ................................. 48
Figure 15: Help Menu ................................. 52
Figure 16: Build SQL Dialog ............................. 53
Figure 17: Microsoft Notepad w/ SQL File ............... 56
ABSTRACT

This paper explores the possibility of coupling SQL with a semantic data model. For this study, the primary objective was to build a working prototype of a program that allows a database designer to define data objects and their respective interrelationships using the Object-oriented Semantic Association Model (OSAM*).

The prototype isolates from the designer the low level commands (i.e., CREATE TABLE, CREATE INDEX) which comprise the SQL data definition language (DDL). Once the objects are defined by the designer, the prototype generates the relational database table definitions without the designer having to directly use the SQL DDL.
Chapter 1
INTRODUCTION

Of the myriad of applications in which computers have been employed, perhaps none is so pervasive as the database. Since the advent of computers, an ever-growing body of work has been compiled with regard to database management in both industrial and academic circles. Not surprisingly, a considerable amount of controversy has never been far behind. Today, the whole question of current and future trends in database management is one that is hotly debated—particularly in the area of relational database technology.

The fundamental concepts of the relational model were first documented in the early 1970s [Codd70]. A few years later, the original Structured Query Language (SQL) was formally presented as a language for the defining and manipulating of a relational database [Cham74]. Since that time, SQL driven relational databases have evolved into a de facto standard and have been widely accepted in the industry. Even so, SQL driven databases have been heavily criticized for their inability to adequately represent more complex relationships [Codd79]. For this reason, other database methodologies continue to be researched.
One such methodology that is especially prominent in the literature is that of semantic data modelling. Semantic data modelling grew out of the need to more easily depict data abstractions that more closely resembled real world objects. The first published semantic model appeared in the mid-1970s [Abri74]. This and other early papers on the subject presented semantic models as strictly a high level tool for initial schema design, rather than as a full fledged database management system (DBMS). Some of these earlier models include the Entity-Relationship (ER) model [Chen76], the Functional Data Model (FDM) [Kers76], and the Semantic Database Model (SDM) [Hamm81]. Progressively, these semantic data models became more refined and the scope extended to incorporate a database structure. In recent years, a renewed interest in semantic data models has been kindled in part by the heightened industry awareness of object-oriented techniques. Modern semantic models, such as the Object-oriented Semantic Association Model (OSAM*) [Su88], and the IRIS model [Derr85], are testimony to this interest.

There is an immediate problem, however, with semantic models coupled with databases in that they are neither standard nor accepted within the industry. Most do make use of a DBMS that is relational, though these databases frequently exhibit nonstandard features. In addition, many of these model driven databases have their own special languages that
do not conform to the SQL standard. This lack of compatibility with what is now a standard will undoubtedly slow the acceptance of such models, however good, by industry.

This is not to say that the SQL driven relational database is the best way to accommodate objects and abstract data types. Rather it is to say that, for better or worse, SQL is a standard and will likely continue to be for quite some time. It may be that SQL just cannot be made to efficiently manage the complex objects that are addressed by these semantic data models. However, if some of the ideas that have been put forth in these semantic models can somehow be incorporated into SQL, the industry would be in a much better position to put to work the research that has been done thus far.

Curiously enough, there appears to be little in the way of literature that pursues the idea of taking SQL as it now stands and adding to it some object-oriented capabilities. With the notable exception of the IRIS model, no other major semantic data models even consider SQL a going concern, preferring instead to specify their own languages. What the literature does provide though, is a good solid framework upon which an object-oriented SQL extension can be built.
There is little doubt that the representing of objects within a SQL driven relational database will have its problems--primarily because relational databases were never designed with that in mind. In some instances, these object-oriented "extensions" may turn out to be very superficial changes to SQL. Nevertheless, the idea is well worth considering for the aforementioned reasons.
For this study, the primary objective was to build a working prototype of a program that allows a database designer to define data objects and their respective interrelationships using a high level, object-oriented semantic model. For the purposes of this study, an object can be defined as follows:

Objects are abstract or atomic entities which correspond to things in the application environment being represented in the database, and may be at various levels of abstraction and of various modalities (media). [McLe91]

The prototype would seek to hide from the designer the low level commands (i.e., CREATE TABLE, CREATE INDEX) which comprise the SQL data definition language (DDL). Once the objects are defined by the user, the prototype would have the ability to generate the relational database table definitions without the designer having to directly use the SQL DDL.

2.1 Selecting a Semantic Data Model

Rather than define a semantic model expressly for this project, an existing semantic model--the Object-oriented Semantic Association Model (OSAM*)--was selected as the
basis for the prototype. OSAM* was selected because it is one of the foremost object-oriented models in the literature as well as one of the most accessible. The prototype encompasses only the fundamental principles of OSAM*, as laid out by Su [Su88]. These principles as they relate to the project are outlined briefly here:

As with many popular semantic models, OSAM* makes heavy use of graphical depictions of objects and their interrelationships. In OSAM*, these depictions are referred to as semantic (S-)diagrams (see Figure 1) that consist of a series of nodes which are linked together by lines. Each node represents an object class.

![Figure 1: A Simple S-Diagram](image)

In OSAM*, there are two major classes of objects, one of which is the domain (D-)class. According to Su:

A D-class specifies a domain of permissible values over which attributes of other classes can be defined. [Su88]
A D-class object can represent a kind of data type, such as a string, number, or boolean, or may represent an actual set of values. In the context of an S-diagram, domains are depicted as circles that may be connected to other objects which have attributes dependent upon that domain.

The other object class in OSAM* is the entity (E-)class of objects. According to Su:

An E-class represents a set of objects that correspond to entities in an application domain. [Su88]

Every instance of an E-class is defined and identified by a set of attributes. These attributes, in turn, may be inherited by other E-classes, and mapped to values in a predefined D-class.

In addition to these two classes, there are five semantic associations for these object classes in OSAM* that are discussed in [Su88]. These five associations derive their definitions from the different interrelationships allowed among E-classes and D-classes and are represented within the S-diagram as the lines that link the nodes together.

From the start, the prototype was envisioned as providing a graphical representation of OSAM* objects based on the OSAM* S-diagram. Because of the emphasis on the graphical aspects of OSAM*, it seemed logical that the prototype would be most
effective if it were built to run within a graphical user interface (GUI). Here again, because the focus of the project was not the GUI itself, an existing GUI would be selected and the prototype built to run within it rather than building a GUI as part of the prototype. With ease of use in mind, the prototype would be made to work in a way that would be consistent with other programs that run under the GUI, adhering as much as possible to any existing standards within the GUI.

2.2 Selection of tools for development

In selecting the host GUI to be used in the project, the primary consideration was industry acceptance. Since the idea was to build a prototype with the industry in mind, the host GUI had to be something of an industry standard. The second major criterion was the physical availability of the GUI and the software required for applications development. In the end, the Microsoft® Windows™ 3.1 GUI (hereinafter referred to as "Windows") was chosen.

With respect to the developer tool to be used to build the prototype, ease of learning and ease of use were the two major criteria. As with any other GUI, Windows has its share of low level programming concerns such as memory allocation, resource compilation, and message handling which must ordinarily be addressed by a programmer using a
traditional language like C, or Pascal. Since these Windows specifics can be difficult to master, attention was focused on those packages which isolate the developer from the lower levels of Windows programming. Strongest consideration was given to those packages that offered an Integrated Development Environment (IDE) within Windows so that the user could actually develop, compile, run, and debug the code from within Windows. Of the packages that were evaluated, Gupta Technologies® SQLWindows™ package (hereinafter referred to as "SQLWindows") was finally selected. In addition to meeting the aforementioned requirements, the package also comes with a single user SQL database that runs under Windows, and an entire suite of SQL functions for accessing the database.

Of course, the very strength of SQLWindows would also prove to be its major weakness. Since the package does isolate you somewhat from the lower level Windows functions, it is not always easy to employ lower level Windows calls from within your code when you need them. Secondly, the product is meant to be used for designing database applications—which this prototype is not. Specifically, it was noticed that the actual graphical capabilities afforded by the package were extremely limited. There are no SQLWindows functions for dynamically drawing lines, boxes, or circles on the basis of any kind of a coordinate system. The other major limitation was the lack of internal data structures
toward working with a SQL database, it is assumed that all complex data structures are somehow represented and managed within the database. Only the single dimensional array is native to SQLWindows. Advanced data structures such as multi-dimensional arrays, link lists, binary trees, and pointers are not supported. This fact was of much greater concern than the lack of graphical capability, because the primary focus of the prototype was not graphics--but affording a means of generating SQL from a semantic model depicted in part by graphics. If the intent was to build a commercially viable product, the typical abilities of panning, zooming, and printing graphics would have to be present; but for the purposes of the prototype, nominal graphics capabilities were required.

2.3 Understanding OSAM*  

Once the development platform was determined, the underlying issues and design for the prototype could be finalized. The two most important considerations in designing an OSAM* based tool of this nature are: 1) how an OSAM* defined schema is to be translated into an SQL database, and 2) how this schema is to be presented to the end user.

With respect to the first item, it is submitted that the basic structures found in any given OSAM* schema can be represented in a SQL database using standard SQL tables and
views. Elements of OSAM* that cannot readily be accommodated by SQL can be captured by the prototype and, at the user's option, stored in a special SQL table as metadata to be referenced and updated by the programmer(s) throughout the development life cycle.

To better understand how these tables are derived, consider a typical OSAM* s-diagram. In OSAM*, there are two basic classes of objects that can be modeled. The first class is the Entity, or E-Class. Entities are represented in an s-diagram as a rectangle and can reference any other element within the OSAM* model as attributes of itself. The way a particular entity references another element is known as its semantic association. The five different semantic associations defined in OSAM* are generalization, aggregation, interaction, composition, and cross product which are denoted in an S-diagram by the letters G,A,I,C, and X respectively.

The other OSAM* class is the Domain, or D-Class. Domains are depicted in an s-diagram as a circle and may either be defined as simple or composite. Simple domains usually are defined in terms of elementary data types, such as DATE, INTEGER, or CHAR. They may consist of a finite set of values, or a range of values. They may not make reference to any other entities or domains defined in the model. Composite domains are defined in terms of other classes and
are usually composed of two or more simple domains, though other entities may also be referenced within the model.

Starting with the OSAM* s-diagram, the first step was to identify those aspects of OSAM* which either corresponded directly with existing SQL structures, or could be easily made to do so. In so doing, several less obvious structures within OSAM* were uncovered. Much of the semantic model lends itself very nicely to SQL with some minor modifications. The following represents a summary of the OSAM* structures, or objects, that are implemented in the prototype:

- Simple domains as defined in OSAM* are depicted in SQL as elementary data types. There is no good way to represent some of the OSAM* simple domain structures, such as the incorporation of functions (i.e. COMPUTE) or restrictions on number or date fields into a SQL data type, so these aspects of OSAM* were left as part of the SQL metadata structure.

- OSAM* composite domains are represented in SQL as a simple lookup table. By definition, a composite domain usually is a finite group of values with which other OSAM* objects may be associated. Domains can inherit attributes only from other domains. Objects associated
with composite domains inherit any attributes used to identify the domain as part of the object.

- **OSAM** regular entities are represented in SQL as a regular data table, complete with indexes and primary keys. In OSAM*, regular entities can provide attributes to other regular entities with which they are associated. Domains may be inherited by regular entities.

- Composite entities are represented via SQL views. In OSAM*, a composite entity consists of summary attributes derived from a dynamic group of objects. Within OSAM*, composite entities may be nested to any level (a composite of multiple composite entities can be created), but within the prototype only one level of composites is permitted. Composite entities, by definition, are derived from the regular entities with which they are associated, and may have no attributes of their own.

- Like composite entities, cross product entities are defined via summary attributes from other objects, and also take the form of a SQL view. However, cross product entities may have attributes of their own which determine the means by which summary data is broken out. Cross product entities get their attributes from one or more predefined domains with which they are associated. When a regular entity is associated with a cross product...
entity, all domains inherited by the cross product entity are passed on to the regular entity.
Once the correspondence between these OSAM* objects and SQL structures was defined, the next step was to design the user interface. Of primary concern when designing the overall presentation of an OSAM* s-diagram was determining those aspects of the semantic model that would be represented graphically.

3.1 OSAM* Concerns

As is the case with most semantic models, the primary difficulty encountered when working with OSAM* is in depicting a given logical database schema graphically. In all but the simplest systems, graphical representations quickly become unmanageable and an alternative means of documenting the database design becomes necessary.

Instead of trying to keep up with all levels of detail within a given database schema, it was decided that the user be provided a high level depiction of the entire system in which only the major system entities would be shown. From here, the user would be given the option to create or remove high level entities from the diagram and associate them with one another via connecting lines. Also, the prototype had
to allow the user to be able to place or move (cut and paste) objects on the screen. It was also desired that the prototype support some simple rerouting logic as objects were placed on the screen.

Because of the limited graphics capabilities of SQLWindows, programming the graphical portions of the prototype was something of a challenge. To an end user, the prototype had to be able to draw boxes, circles, and lines much in the way that an OSAM* s-diagram would appear. Unfortunately, there are no functions internal to SQLWindows that allow a programmer to do this. In SQLWindows, lines must be hard-coded objects defined within a top level window and are generally used to give an aesthetic touch to the interface. Lines cannot be programmed to respond to mouse clicks or any other user actions. They cannot be generated dynamically unless the program accesses special graphics functions in the Windows API. Microsoft does manufacture a Windows Software Development Kit (SDK) for the purpose of using the API functions; but at the time, this was unavailable.

Eventually, special functions were written to keep track of the window handle assigned to each line. A window handle is an internal identifier which is assigned to any given Windows object at the time that object is created. Given a window handle, SQLWindows does have functions that allow a programmer the ability to hide and show windows, thus
creating the illusion that the line is "drawn." The logic that dictates which lines are drawn is hard coded based on which OSAM* objects on a screen are associated with one another. However, since each line is its own Windows object, too many lines defined on a given top level window can affect the performance of the program. For every OSAM* object depicted on a screen, a whole set of lines had to be hard coded to allow that object to be "associated" with any other object on that screen. This served to be the primary limitation as to how many OSAM* objects could be depicted on a screen. To accommodate 10 OSAM* objects on the screen, 62 separate lines were required.

Boxes and circles were somewhat easier to implement. Like the lines, all the boxes and circles had to be predefined objects on the top level window. Boxes were represented by data fields (which are rectangular by default) and circles were represented by picture fields, which are used for displaying bitmaps on a window. Fortunately, SQLWindows has a considerably richer set of functions for manipulating data and picture fields. Data fields are used to display numbers or text and can interpret and take actions on a Windows SetFocus message, which is received whenever the user tabs into the data field or clicks on it with a mouse. Picture fields can also intercept a mouse click or double click.
3.2 SQL Concerns

With respect to SQL, the principle design issue was determining the extent to which the user would be isolated from SQL itself. Ideally, the semantic model on the screen would be as complete a schema as would be needed to generate a complete set of SQL statements, but as with most semantic data models, OSAM* was not designed around SQL. OSAM* is best at modeling the overall relationships between objects which roughly translate to relationships between tables in SQL. However there are certain aspects of SQL, such as the length of a character field within a table, which cannot be practically designated on a traditional OSAM* s-diagram. At some point, then, the user would have to be required to key in all of the detail necessary for SQL; so the original intent was for the prototype to isolate the user from as much of the SQL syntax as possible and provide an easy means for entering the required SQL details. The prototype could then generate the SQL for the s-diagram, and write the finished SQL to a file.

Another major concern regarding SQL was the question of referential integrity. Until relatively recently, the concept of referential integrity (rules that guarantee the validity and accuracy of tables which are reliant upon other tables within the database) was the exclusive domain of relational database theory since few, if any, commercially
available databases actually supported it. As of this writing, there are several vendors that offer databases that provide referential integrity. For its part, SQL has always had the PRIMARY KEY and FOREIGN KEY constructs for defining referential integrity among its tables, but there are still a number of databases that, while they profess to be SQL databases, do not recognize these clauses. Since it was a primary objective that this prototype be useful across any SQL database, it was felt that it should be able to generate PRIMARY KEY and FOREIGN KEY clauses at the user's option.

Finally, there was the question of the SQL syntax itself as it tends to vary from database to database. Many vendors offer enhancements to SQL in their products which are not ANSI SQL compliant. For this reason, it was decided that the prototype would generate ANSI SQL only, on the assumption that most SQL databases should support at least the ANSI standard.
Chapter 4
IMPLEMENTING THE OSAM* DESIGNER

4.1 The Workspace: Showing the s-diagram

It was desired that the prototype's main window present the user with an overall workspace within which the user would construct a likeness of an OSAM* s-diagram (see Figure 2). The workspace is limited in the number of objects that could be viewed on any one screen, but the user has the ability to scroll horizontally and vertically around the workspace. For the purposes of the prototype, the number of screens allowed is limited to 324 arranged in an 18x18 matrix; such that from the center of the workspace, the user may go 9 screens in any direction. At all times, the user is able to see their relative position within the entire workspace via screen coordinates displayed at the lower right hand corner of the workspace. Screen (0,0) denotes the center of the workspace, whereas screens having horizontal or vertical coordinates of 9 or -9 denote the boundaries of the workspace. These screens hold up to 10 objects apiece, but are also designed to partially overlap one another when scrolling such that any individual screen can only hold five distinct objects on average. In all, well over 1500 objects
can be accommodated within any one s-diagram, where a single object usually corresponds to a SQL table.

Figure 2: Main Window

Key to maneuvering around the workspace and working with objects is the concept of object focus—the idea that at any given time within the workspace, a single object has focus and only that object may be worked with. In the prototype, the user is able to set focus to any object visible on the screen via the keyboard or the mouse. An object with focus is depicted in the workspace as yellow instead of the
normal, light blue color for all other objects. In addition to the coloring, data identifying the object with focus is displayed in the lower part of the main screen. This is important because the object which has focus does not necessarily have to be visible on the screen because focus on a given object is not lost as the user scrolls around the workspace.

4.1.1 Representing of OSAM* Objects

Upon starting the program, the user begins in the center of an empty workspace. At this point, the user is only allowed to create objects (unless the user opts to load a previously saved s-diagram file). To add an object, the user would pull down the "Object" menu (see Figure 3), and select the "Create..." option. The prototype then presents the Add Object dialog box (see Figure 4) which prompts the user for an Object ID, name, type, notes, and a location.

Figure 3: Object Menu
In OSAM*, every object is assigned an Object ID, or OID, which uniquely identifies that object from all others within the s-diagram. It is unclear whether the OID as defined in [Su88] is supposed to be a number, or if it can be a label. Within the prototype, the OID is defined as an alphanumeric field that can be up to 7 characters long. The OID here serves the same purpose as within OSAM*, as the user will refer to the OID whenever working with that object. It is the OID which actually labels the object within the workspace, and identifies the object in all the Windows dialog boxes, or dialogs. At the time an OID is entered, the prototype verifies that it is a unique identifier and that it is not a SQL reserved word. (A list of SQL reserved
words is read in from a file and stored in an array when the
program first starts.)

Since an OSAM* object usually corresponds to a SQL table,
the prototype requires a name for any object created. The
sole purpose of the name is to provide a name for the SQL
table at the time the SQL statements are actually generated.

There is nothing wrong with using the OID as the table name;
and, the table name will default to the OID when an object
is first created. However, the OID is limited to 7
characters whereas a SQL table name can be up to 18
characters long. It is important to note that summary
entities do not result in SQL tables being created, and so
the object names for these objects have no function.

However, the prototype still requires that an object name be
provided as a further means of identifying the object. Like
the OID, the name must be unique from all other names, and
must not be a SQL reserved word.

When an object is first created, it must be defined as a
certain type, of which there are four kinds in the
prototype: regular entities, composite domains, composite
entities, and cross product entities. To assign a specific
type to the new object, the user selects the desired type
from a drop down list box. The object type dictates the
kinds of attributes an object may have and the kind of
associations that can be made with it. It is the only
object parameter that cannot be changed once the object is created; an object accidentally created with the wrong object type must be deleted and reentered.

The prototype also allows the user to attach a note which may be up to 254 characters long to any object. It is not required for any object. The note itself serves no purpose within the prototype itself, but may be stored as a remark in the SYSTABLES table when the user builds the SQL.

Finally the user must indicate an object location when adding an object. The location specifies where the object is to be drawn on the screen. The user may select any location on the visible screen that is not currently occupied by another object; and is not allowed to create objects at all on screens that are full. Once the object is defined, it is immediately placed in the workspace. In OSAM*, the s-diagram depicts any entity as a rectangle and any domain as a circle, and it was initially planned that the prototype should do the same thing. Within the prototype, however, it was not immediately clear how to create a circle that could contain a label (the OID) with SQLWindows. As a result, an entity in the workspace is represented by a light blue rectangle; whereas the domain is shown as a gray rectangle with a small, light blue circle either above or below it. As focus is set to an entity, the
rectangle turns yellow; with domains, the rectangle remains gray and the circle turns yellow.

Once added, objects within the workspace may be associated with one another. To associate an object with another object, the user would pull down the "Object" menu, and select the "Associate..." option. At that time, the object which currently has focus becomes the host object and the prototype then presents the user with the Associate Object dialog box (see Figure 5). The Associate Object dialog presents the user with a drop down list of the different object types that may be associated with the host object. If the host object is a composite domain, it may be associated with regular entities, cross product entities, or other domains; otherwise it may only be associated with regular entities. When the user selects a given object type to be associated with the host, the prototype lists all objects of that type that are not already associated with the host object in a separate list box from which the user may select the object that is to be associated with the host. Whatever object the user selects from this list becomes the dependent object.
At this point, the user must select the type of association that will exist between the host and the designated dependent object from another drop down list box. Like the object types list, the allowable types of association are dependent on the type of the host object; as well as that of the selected object. At any point when the user has designated an object and has selected the type of association, the prototype presents the user with a brief description of the association that they are about to create. This way, when the user wishes to associate the object DOCTOR with the host object STAFF, the user will see "The DOCTOR is a kind of STAFF..." rather than having to know what a generalization (or "G") relation is. Note that the user is not given the option to specify a G-relation which states that "the STAFF is a kind of DOCTOR..." because the STAFF is the host object, and the DOCTOR is the dependent object. If the converse is desired, then the user
would be required to first set focus to DOCTOR and then associate it with STAFF.

The user may undo any association made by selecting the "Disassociate" option under the Object menu; at which point, the prototype displays the Disassociate Object Dialog (see Figure 6). This dialog presents the user with a drop down list of all objects dependent upon the object with focus. When the user selects a given object in the drop down list, the association type and description fields are refreshed to indicate the kind of association that is about to be undone.

![Disassociate An Object From DOCTOR](image)

**Figure 6: Disassociate An Object Dialog**

In OSAM*, as with many other semantic data models, associations between objects are depicted by connecting the two objects with a directional line or arrow, where the object to which the arrow is pointing is the dependent object. Next to the line would appear a symbol (i.e., "G")
for Generalization, "A" for Aggregation) denoting the kind of association. This posed two major hurdles for the prototype. First, how would the prototype represent relationships between two objects that were not physically located on the same screen. Since there was no way to pan out, a user would not be able to make out such a relationship connected by a line without having to scroll through several screens. Should the lines cross with other lines that span multiple screens, it is likely that a user could be confused. Add to this the fact that the prototype has a refresh rate of between two and three seconds as one pages through screens, and this quickly becomes an infeasible solution.

It was eventually decided that there would be two ways to graphically depict a relationship dependent upon the proximity of the associated objects. If the associated objects were on the same screen, the prototype would connect the two with a line just as would be done in OSAM*. However, associations between objects on different screens would be represented not by lines, but by color. Bearing in mind that at any one time a single object would always have focus, all objects associated with the object having focus that were not on the same screen would be shaded green, rather than blue. The user would not need to see the focus object on the screen because the focus object's OID and other information are always displayed at the bottom of the
main screen, so the user always knows which object has focus. However, this solution still didn’t solve the screen problem, as the user would still have to page through the screens looking for green objects. To keep screen scrolling to a minimum, a "Find..." option was added to "Object" menu that provided the user with a Find Object dialog (see Figure 7). From here, the user would be able to list all objects of a given type, or objects associated with the focus object--then jump directly to the screen having a particular object.

![OSAM* Object List](image)

**Figure 7: Find An Object Dialog**

The second major issue with the OSAM* association was with the arrows that indicated the direction of the association. Given the amount of code required just to show lines on the screen, it was clear that showing arrows would not be feasible; and lines couldn’t be used at all for associated objects on different screens anyway. What was finally done
was to use the symbol denoting the kind of association as
directional indicators—rather than an arrow. Whenever an
object received focus, all other objects associated with
that object would have the appropriate association symbol
(i.e., "G", "A") appear to the right of the OID. If the
associated object was dependent on the object with focus,
then the symbol would appear in lowercase. Conversely, if
the object with focus was dependent upon the associated
object, then the symbol would be capitalized. The symbols
themselves would appear only on objects associated with the
object with focus, and would disappear (or be refreshed)
whenever the focus changed.

Once the logic for the drawing of lines, boxes, and circles
was tested, the next major hurdle was the representation of
the workspace. It had been decided that the workspace
should let the user scroll in any direction; and that the
scroll would be partial such that a portion of the previous
screen would appear on an adjoining screen. This was the
first of many instances where having only single dimensional
arrays made for a challenge.

The scrolling logic would not have been so bad were it not
for the partial scrolling; but the partial scrolling was
viewed as necessary not only for the user to keep their
bearings, but also to allow multiple views of any given
object so that an object can visually be associated with
more objects. Otherwise, the user would only be able to see an object on one screen, and could at most visually associate that object with nine others. With the partial scroll, the user would be able to see side objects on two screens and corner objects on four, thus allowing the user to view up to 36 different associations (see Figure 8).

![Figure 8: Workspace "Cells"](image)

A special array is defined in which each array element represented a single location, or cell, in the workspace. Starting from the upper left hand corner screen, each of the screen cells would be assigned a sequential number. These numbers could then be assigned to objects and the program would be able to determine the location of any given object on the screen based on that number. Without the partial scroll, it would have been easy to key the first 10 objects on the first screen as the first 10 elements of the array, the 10 objects on the next screen as elements 11 through 20, and so on. Every object then would be tied to a single location in the workspace. At the time a user created or
moved an object, the program would need only calculate the address of the cell, based on the screen coordinates and the cell’s own location on the screen, and store that number with the object. However, partial scrolling resulted in the placing of the same object on two or four screens and a means for keeping up with multiple addresses per object had to be designed.

The cell array was used to hold object numbers that would be generated sequentially starting at 1 and assigned to objects as objects were created. Within the program, these object numbers would serve to identify the location of a given object’s OID, name, type, and description—each of which would be held in their own global arrays. Therefore cells 10, 17, and 64 might contain the object number 23; in which case OID(23) would hold the object’s OID.

Once the cell array was in place, the logic for maintaining the array had to be designed. Looking at Figure 8, if the user added an object and associated it with position 6, the program would have to be smart enough to figure out that position 15 would have to be updated to contain that same object. To accommodate this, the prototype has to calculate the affected positions on other screens based on a position that has been changed or added to. For example, if an object is added to the sixth position on any screen, the program adds 9 to the corresponding array element’s address
to come up with the address of the affected element (or cell) and set that cell to the same object unless the screen is on the right-hand border of the workspace, in which case no other cells are changed.

After the partial scroll had been designed and built, the next hurdle was in handling the associations. Unlike the OID, name, and other object parameters captured at the time the object was added, a single object can, and typically will, have multiple relationships with other objects. However, the amount of information that needs to be carried by any one association is minimal. All that is really needed is the number of the associated object, the type of association (i.e., generalization, aggregation), and the direction of the association. It was decided that it was needless to create separate structures for each of these pieces of information, and so the two were combined into a single global array. What any given element in the array would hold would be a comma separated string that contained special delimiters for object numbers, and delimiters for associations. The result was an association array in which there could only be one array element per object (just like OID and name) but that each element could be parsed out into multiple relationships for that object. The direction of the association did not need to be stored separately at all because it would be derived based on the association being capitalized or in lowercase. Once these associations were
working in conjunction with the objects and lines being drawn on the screen, the bulk of the OSAM* s-diagram was completed.

4.2 Accommodating SQL in the prototype

The prototype was now to the point where there was little more that could be represented within OSAM*, and designs for the capture and upkeep of SQL data had to be planned. It was clear at this point that the SQL data, or object attributes, would have to be stored in a separate structure than that of the OID, associations, and the other object parameters; but that these attributes would somehow have to be referenced by the object itself. In addition, this structure had to be completely dynamic, allowing the user to add, delete, or change individual attributes. Finally, the structure would have to hold enough information about the attribute to construct a SQL table with it. It would also have to hold any information concerning the creation of primary keys and indices in SQL.
4.2.1 Representation of OSAM* attributes

There was no question that this structure would eventually become implemented as an array, or more likely, a series of arrays. The problem was with the need for quick access. It would have been fairly simple to have done a brute force traversal of an array holding the object numbers looking for just those attributes having a given object number until reaching the end of the array; but in a large OSAM* definition with hundreds of tables, this method might not be acceptable.

What finally had to be done for performance purposes was to emulate a series of link lists using an array. An element within the array that marked the start of a particular link list for some object would have its address stored as a parameter of the object itself. Objects having no attributes, or objects just created would have this address set to zero. Each of the individual attributes stored for that object would also store a pointer that the program could use to jump directly to the next attribute, and so on until the next pointer came up with a sentinel value (-1). Given a particular object, the program could then retrieve the object parameter that pointed to the start of the link list, jump to that element, and start traversing the list. Naturally, as attributes were added and deleted from objects, the program would have to clean up the links.
The link list worked fine for just representing those attributes that were defined under a given object; but how would the structure be able to handle attributes inherited from objects associated with a given object? Referring back to a previous example, if DOCTOR is a kind of STAFF, and every STAFF is identified by the attribute LICENSE, then every DOCTOR must have a LICENSE. The problem is that LICENSE is defined under the STAFF, not the DOCTOR. Somehow, then, this structure had to be equipped to handle inherited attributes.

The first approach taken to handle inherited attributes called for maintaining an image of every object in the system, with each image being its own link list in the attributes structure. Therefore, if there were twelve different objects that were a kind of STAFF, then the attribute LICENSE would be duplicated in twelve different lists. It was felt that it would be easier to traverse the OSAM* structure for any one object and reconstruct its attribute list at the time that changes were made, rather than construct all objects based on the associations and their own attributes at the time that the SQL was generated.

There is no easy way to solve the problem of inherited attributes; but maintaining an image of every object never worked quite right from the start, and ultimately had to be
abandoned. The most difficult problem arose when tracking attributes across dependent objects. For example, unless an association between two objects is defined as a generalization (or G-relation), only key attributes are inherited by any dependent objects. So, if a user changed the key of an object, all of the keys inherited on through the s-diagram would have to be changed as well. For their part, G-relations are even more complex because they can go as deep as the user desires, and the attributes defined in each object involved have to trickle all the way down to all objects associated directly or indirectly. For instance, an INTERN may be defined as a kind of DOCTOR, in which case INTERN would inherit all the attributes from both DOCTOR and STAFF.

The other major consideration with tracking attributes is handling the case where the user creates a duplicate attribute. One of the many rules in SQL is that, for any one table, no two columns defined for that table may share the same name. Otherwise, SQL would not know which column is being referred to when a database transaction is being performed on that table. The prototype has to be able to reliably loop through all associated dependent objects of a given object, as well as the given object itself, and verify that this attribute is not duplicated. For example if INTERN had an attribute called SPECIALTY, then the user
would not be able to enter SPECIALTY in either DOCTOR or STAFF.

Duplicate attributes can also be encountered when the user creates associations. Had the user entered in the SPECIALTY attribute for both DOCTOR and INTERN prior to creating a G-relation between the two objects, it would be the responsibility of the prototype to catch the duplicate SPECIALTY attribute that would result in INTERN at the time the G-relation is created. Akin to this problem is dealing with OSAM* associations that are altogether invalid. A good example of this is when the user attempts to create an association that results in a loop (i.e., an INTERN is a kind of DOCTOR is a kind of STAFF is a kind of INTERN).

This is clearly an invalid association; but if this associative loop is created prior to any attributes being keyed in, the logic which checks for duplicates would not catch it.

As it turned out, the underlying logic that would have to be built to handle any of these duplication errors was completely dependent upon the ability to reliably traverse the OSAM* definition in real time. Once it was determined that this piece was an absolute, it became unnecessary to maintain separate attribute lists for each object. Special functions were written that would return a list of ancestor objects (objects from which a given object inherited
attributes) or dependent attributes (objects dependent upon a given object for attributes). For associations, special functions called IsAncestor() and IsDependent() were written so that objects either directly or indirectly associated with the host object would not be brought up in the Associate Object dialog.

While all of this design and coding transpired, the user interface for keying in all of the SQL details gradually took shape. The finished Object Attributes dialog (see Figure 9) presented the user with a large table window, in which each row in the table window represented an object attribute. In the window, the user can see all attributes defined for that object shown in black, and all inherited attributes from associations with other objects in green. The dialog allows the user to freely change or remove any attribute shown in black. Since objects shown in green are inherited, they cannot be changed and cannot be removed without destroying the association that caused the attributes to be inherited in the first place. Finally if an object, through association, requires attributes from another object which have not yet been defined for that other object, an "(undefined)" entry is shown in red in the table window. To correct this problem, the user must either define attributes for the other object, or disassociate the two objects entirely. If this is not corrected, the
prototype will produce an error message when the user attempts to generate SQL for the object.

<table>
<thead>
<tr>
<th>Name</th>
<th>From</th>
<th>Assoc</th>
<th>Key</th>
<th>Req</th>
<th>Index</th>
<th>Type (Press spacebar to show)</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td>30 Character Field</td>
</tr>
<tr>
<td>height</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>Decimal (±99)</td>
</tr>
<tr>
<td>weight</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>Decimal (±99)</td>
</tr>
<tr>
<td>sex</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>1 Character Field</td>
</tr>
<tr>
<td>blood_type</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td>2 Character Field</td>
</tr>
<tr>
<td>condition</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td>2 Character Field</td>
</tr>
<tr>
<td>date_admitted</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>Date</td>
</tr>
<tr>
<td>date_released</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>Date</td>
</tr>
<tr>
<td>comments</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td></td>
<td></td>
<td>Long Character Field</td>
</tr>
<tr>
<td>last_updated</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td>System Time Stamp</td>
</tr>
<tr>
<td>nurse_ssn</td>
<td>NURSE</td>
<td>M</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>9 Character Field</td>
</tr>
<tr>
<td>floor</td>
<td>ROOM</td>
<td>A</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Decimal (±9)</td>
</tr>
<tr>
<td>room</td>
<td>ROOM</td>
<td>A</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Decimal (±999)</td>
</tr>
</tbody>
</table>

Figure 9: Object Attributes Dialog

When a new attribute is added to an object, the user must provide all the details necessary to properly define a SQL table and any indices. In essence, object attributes correspond directly to a SQL column definition within a table. In designing the interface of the Object Attributes dialog, the primary objective was to isolate the user as much as possible from SQL syntax. Also, since a large number of attributes can be defined for any given object, efforts were made to minimize the number of keystrokes necessary to actually enter all SQL details for any one
defined in the prototype in about 10 keystrokes after the attribute is given a name which may be up to 18 characters long. The following information is captured for any attribute entered.

- **Name:** Every attribute must have a name. The attribute name is ported directly into SQL as the name of the corresponding SQL column. Names may be up to 18 characters long.

- **Key:** denotes whether an attribute is used to identify an object instance. In SQL, key attributes will translate into unique indices and primary keys.

- **Required:** denotes whether a value must be assigned an attribute in every instance of an object. In SQL, required attributes will become columns defined as NOT NULL. Key attributes are always required.

- **Indexed:** denotes whether an attribute should be indexed for purposes of data retrieval. In SQL, required attributes will translate into regular indices. Key attributes are always indexed.

- **Type:** represents the data type of an attribute (i.e., character, number, date). In SQL, a data type must always be defined with a column at the time a table is
always be defined with a column at the time a table is created. Allowable types from which the user may choose may be displayed when the user moves the cursor to the "Type" column, and presses the spacebar.

- Description: a short description of the attribute as it relates to the parent object. It is stored as a remark within the SYSCOLUMNS table, and is an optional field.

In addition to these basic attribute definitions, summary attributes may be defined for individual compositions or cross products. Recall that compositions and cross products take the form of special entities, called summary entities, which may be associated with any regular entities. When such an association is made, summary information can then be defined for that summary entity based on numeric attributes defined for the regular entity. To define a composition or cross product for a regular entity, the user is provided with two drop down list boxes on the Object Attributes dialog which contain all composite and cross product entities to which the object is associated. When the user selects one or the other, a list of all numeric attributes defined for that object is presented in the Summary Attributes dialog (see Figure 10). Next to each attribute are the four standard aggregate functions supported in ANSI SQL: Lowest (MIN), Highest (MAX), Average (AVG), and Total (SUM). The user may then go through each of the numeric
attributes and selectively enable or disable each of these functions. When the user has defined each of the summary attributes within the entity, the prototype saves the composite or cross product definition as part of the overall entity.

Once the user has finished defining the attributes for any one object, the prototype then begins a series of error sequences to validate that no attribute names are duplicated. It was found that the easiest way to do this was to actually retain a snapshot of the object as it was immediately prior to its attributes being changed. At that time, all changes made to the attributes are saved to the

Figure 10: Summary Attributes Dialog
object or a name that conflicts with the list of SQL reserved words, are trapped without creating a snapshot.

The first thing that happens at the time attribute changes are saved is that all objects which are dependent upon the changed object, their respective associations, and the level of removal are read into separate arrays. The level of removal is dictated by the number of dependents that constitute an actual association. For example, DOCTOR is only one level removed from STAFF; whereas INTERN is two levels removed since INTERN is dependent upon DOCTOR for its association with STAFF. At that point, each of the dependents' attributes are traversed separately with the new definition to ensure that no conflicting names occur in any of the objects. In the event that a duplicate is found, the user is alerted to the problem and is given the option to fix the duplicate, or discard all changes just made. The former option results in the Object Attributes being brought up with the just saved changes; whereas the latter results in the snapshot being restored. Though fairly involved, this was viewed as being a much simpler approach as opposed to waiting to create the SQL definitions and, at that point, checking the entire OSAM* definition.

In addition to being able to tie attributes to an object, the user is also given the ability to add rules to regular attributes by selecting "Define Rules" under the Object
In addition to being able to tie attributes to an object, the user is also given the ability to add rules to regular attributes by selecting "Define Rules" under the Object menu. Rules are basically high level comments that may be captured for each object that detail any steps that may have to be taken as an object's corresponding data table is accessed within the SQL database. There are three different kinds of rules supported in the prototype: insert rules, update rules, and delete rules (see Figure 11). At the time a SQL script is generated, the user has the ability to include these rules into a special rules table on the database. This table can then be accessed by the programmer through the course of development.

![Figure 11: Object Rules Dialog](image-url)
4.3 Considerations for Microsoft Windows

In an effort to make the prototype comply with Windows standards, the overall menu structure that is presented on the main window closely follows that of a typical Windows program. The leftmost menu is the File menu (see Figure 12), with which the user may create, open, or save OSAM* files.

![Figure 12: File Menu](image)

The file options themselves all make use of the Windows 3.1 common file dialogs so that the procedures for opening or saving a file are identical to that of other Windows programs. When opening a file, the user has the ability to search for and open OSAM* files having an extension of ".sam" or generated SQL files having an extension of ".sql" (see Figure 13). Finally, the Build SQL and Exit option appears under the File menu.
Following the File menu is the Edit menu (see Figure 14), which provides the user with the ability to Cut, Copy, Paste, and Clear objects from the screen. Like the File menu, it was intended that this menu would follow the Windows standard Edit menu as closely as possible; but a number of instances arose where this was not feasible for the purposes of the prototype.

In Windows, the cut and paste functions are usually used together such that the user will generally cut, or remove, data from the screen and immediately paste it elsewhere—whether the data is a value in a spreadsheet, a phrase in a
text document, or a bitmap. What Windows will do when a user cuts is place the cut data into a behind-the-scenes program called the Clipboard. Later, when the user does a paste, the information currently residing in the clipboard is pasted wherever the user specifies if those data are compatible with where it is being pasted. The issue here is that data, once cut, does not have to be pasted. Cutting data effectively acts as a delete until such time as the cut data are pasted back into the application. If the user never pastes the data, or overwrites the information in the Clipboard with a subsequent cut, then the cut data are gone.

However, one just doesn’t cut an object from an s-diagram as one would a word or phrase from a document; because of the interdependencies involved with that object. The sheer logic that would need to be in place for a program to be able to accurately restore an s-diagram after an object was cut (deleted) and subsequent changes were made went far beyond the original intentions for the prototype; and as such, had to be compromised. In the prototype, performing a cut implies that the user will do a paste. The prototype enforces this by disabling every other menu option for objects once an object is cut except the paste option. This greatly simplifies the underlying logic because now the prototype need never delete a cut object; it just stores the object and any of its attributes in temporary variables.
within the application until it is pasted back into the workspace. As such, the Clipboard is not used at all.

Like the cut and paste operations, the copy operation also had to be modified somewhat to accommodate the rules of OSAM*. In a typical Windows application, a copy implies that an exact copy of the entire data structure has been placed in the Clipboard for pasting. In the context of OSAM*, an exact copy of any given object would technically mean that the object's OID, name, description, attributes, and associations with other objects would all have to be copied. This will not work for two reasons. First, both the OID and name for an object must be unique for that object; so creating a brand new object with the exact parameters would violate the integrity of the s-diagram. An even more difficult scenario arises with respect to the associations. If an object is exactly copied, then all of the attributes defined for and inherited by that object must become part of the new object. In addition to this, the new object would also have all the dependents that the original object had--and each of these dependents will be inheriting the exact same attributes again. Clearly, these duplicate attributes cannot be allowed.

As a result of this, the OSAM* prototype does not make an exact copy of an object when the user invokes the copy option. The only things that are copied are the base
attributes for that object, and the object type. No associations are carried over at all; so all inherited attributes in the original are not reflected in the copy. At the time the copied object is pasted back into the workspace, the prototype prompts the user for a new OID, name, and description.

Finally, the Edit menu allows the user the clear option, which is used to delete an object from the workspace entirely. Usually, a Windows application will provide a clear option coupled with an undo option—in case the user did not mean to clear data from the application. As stated before, however, the ability to totally delete and subsequently restore an object is beyond the scope of the prototype; so the undo option is not supported. Instead, upon selecting clear, the user is warned that they are about to permanently delete the object which has focus, and are given the opportunity at that time to cancel their action.

A Help menu (see Figure 15) and a simple help file has also been included in the prototype to make use of the complete Windows 3.1 help engine. For the purposes of demonstration, however, the prototype will call the Microsoft help file which explains how to use the help engine. At the bottom of the Help menu is a menu item that calls the About dialog, which serves to identify the prototype.
4.4 Generation of SQL from OSAM*

Once the user has completed entering the objects, their individual attributes and associations with other objects, they may build a SQL file. To invoke this function, the user may select "Build SQL..." under the File menu. The prototype will then present the user with a Build SQL dialog (see Figure 16), which provides the user with a number of options for generating the file. A list of these follows:

- Text Format: The finished SQL file will be a readable text file, and this option allows the user to specify the format in which the SQL commands will be written to the file. Choices here are: 1) Uppercase SQL commands only, 2) Uppercase object definitions only, 3) Proper (first letter of each word is capitalized, all other letters lowercase), 4) Uppercase, and 5) Lowercase.
Figure 16: Build SQL Dialog

- **Object List**: Allows the user to limit the SQL build to specific objects instead of generating SQL for everything in the s-diagram.

- **Overwrite SQL**: The user may specify whether the SQL definition will assume new tables, or that the tables have already been defined and need to be overwritten. In the event the latter is selected, the prototype will automatically create "DROP TABLE" statements preceding the "CREATE TABLE" statement.
• Build Rules Table: The user may provide a name for a table which will contain all rules defined by object within the s-diagram.

• Include Comments: Notes and rules entered for each object are displayed as comments throughout the SQL file.

• Referential Integrity: If the SQL database for which the generated file is intended supports referential integrity, the prototype will generate PRIMARY KEY and FOREIGN KEY definitions within the file.

• Store Composite/Cross Product Entities: If the user wants to represent summary entities in the SQL database as views, the prototype will generate CREATE VIEW statements for composite and/or cross product entities within the SQL file.

• Issue a Commit: At the time a SQL definition is created on a database, the definitions themselves are committed, or permanently saved, to the database. Otherwise, the user may opt to rollback, which will cause the database to "undo" these transactions, and restore the database to a previous state. Issuing a commit at the end of the file may save the user from having to manually commit the SQL tables just created from the generated file.
When the user has specified these options on the Build SQL Dialog and continues, the Windows 3.1 Common File Dialog will pop up and allow the user to enter a name to which the generated file will be saved. When a valid file name is entered, the prototype proceeds to generate the SQL file. Each non-summary object included in the build first has its table definition (CREATE TABLE statement) generated. At this point, there are two possible errors which may occur. The first error occurs when the object being tested has no attributes, in which case the build process simply skips the object entirely. The second error results when the object is dependent upon another object for attributes which have not yet been defined in the other object. In this case, the build process will generate an incomplete SQL statement based on the information that is there. Both errors will result in an error message being printed to the SQL file, and the build process will continue. Following the object table definition, any indices required by the SQL table are then written to the SQL file. Finally, each summary definition for that object has a corresponding CREATE VIEW statement generated. Once all object definitions have been created, the build will then create a series of ALTER table...FOREIGN KEY statements to account for referential integrity (if the user has opted to include these statements in the SQL file.) When finished, the prototype automatically brings the SQL file up for viewing in the Microsoft Notepad (see Figure 17).
4.5 Results of the Project

The finished prototype represents the culmination of the project. It demonstrates that a given semantic data model can be incorporated into a program that allows a user familiar with the model to use it to design SQL databases. Specifically, the prototype allows a user who knows nothing
about SQL to generate error-free SQL scripts that are 100% compatible with the database engine against which the generated scripts were tested. Several scripts, including the example cited within this document, were successfully imported into SQLBase 5.0.0, a relational database engine from Gupta Technologies, Inc. The s-diagrams created with the prototype from which these scripts were created addressed all major aspects of the OSAM* model. Together, they encompassed regular entities, composite domains, and summary entities; as well as the five types of OSAM* associations: generalization, aggregation, interaction, composition, and cross products. From these files, SQLBase 5.0.0 was able to create tables for regular entities and domains, views for summary entities, place remarks on generated tables and columns, and create primary and foreign keys on the defined tables based entirely on the semantic model created by the user.

The prototype was also successful in providing the designer a means of maintaining a database schema via a semantic data model rather than having to modify the SQL directly. Any schema built using the prototype can be saved to file and changed at any time. This ability to save and update a schema would serve as an invaluable tool during the early stages of database development. Instead of having to work directly with the SQL statements, the designer need only make changes to the current OSAM* schema, and regenerate a
new SQL script that reflects these changes. Of course, were a developer interested in changing a database definition once the database were actually in production, then the original database would have to be converted to accommodate these changes. If the change were minor enough, a developer in a production environment would probably prefer to make use of the ALTER TABLE statement in SQL and change the database directly. On the other hand, if a major restructuring of one or more tables was to occur, a developer would very likely have to export the data from the affected tables into a flat file. At that point, the developer could go back to the original s-diagram, make the required changes, create the new database schema, and then convert the original data to fit the new schema.
Chapter 5
FUTURE AREAS OF STUDY

The whole task of integrating a complex semantic data model like OSAM* with SQL such that each can make full use of the features of the other is far too broad a scope for this project; and, from the start, this prototype was designed and built with that understanding. What follows are some issues which are prospects for future study.

5.1 Evaluation of Design Techniques

While it was verified that the prototype worked as it was supposed to, tests involving end users working with the finished prototype were not extensive. To fully evaluate the program, it would be necessary to subject the prototype to rigorous testing to at least two categories of database designers--those that know SQL and those who do not, and record and analyze the feedback given by each one of the subjects.
5.2 Improving the Graphical User Interface

There are a number of areas concerning the prototype relating to how it works in the Microsoft Windows environment. In the prototype, the user is allowed to view up to 10 objects at a time on a screen. The program supports printing of an s-diagram one screen at a time, and generated SQL files may be printed from the Microsoft Notepad. In the future, the prototype could be further enhanced by adding options for panning, zooming, and printing the entire workspace.

The edit functions within the prototype could be made to operate in a way which is more consistent with that of other Windows applications. With respect to the cut operation, the prototype requires the user to either paste a cut object, or undo the cut itself. In many Windows applications, it is assumed that a cut object has been deleted from the workspace unless the user pastes it back. One possible enhancement to the existing prototype would be to alter the cut and paste logic so that it could work in this way. To do this, the prototype would have to be able to use an image of the s-diagram prior to when an object is cut or any data required to reconstruct that image, as well as a current view of the s-diagram as it exists after the object is cut. This concept of a before and after image could also be employed in the event the user deletes an
object, and later wishes to undo the delete. Currently, the prototype informs the user that an undo delete is not possible and allows them the opportunity to cancel the delete action.

5.3 Possible Enhancements to SQL

There are some aspects of OSAM* that simply cannot be handled by current implementations of SQL. The OSAM* simple domain, for instance, is far more complex a structure than is its SQL counterpart—the elementary data type. In addition to being able to represent a data type such as CHAR, or DATE, the simple domain may also ascribe certain attributes to that type. For example, any simple domain can be defined as having a specific range of valid values (i.e., between 0 and 60, between 1/1/1980 and 1/1/1990, or must be alphabetic). In SQL, there is no concept of a range. For data of type DECIMAL or type CHAR, SQL does allow the size of the column to be defined, but no control is afforded the user for specifying the content of a column. Simple domains may also be defined as a mathematical expression and in terms of rules, procedures, and functions. In SQL, any such significance attached to a column must be handled programatically.

Similarly, the definition of a SQL table doesn’t quite measure up to that of an OSAM* object. OSAM* allows any
object to have its own set of rules and procedures, which dictate how the object behaves. The prototype approaches this concept by allowing the user to define insert, update, and delete rules for a given object, but these "rules" are simply comments about the object. They are included within the prototype to provide the designer with the ability to record some notes on how each object will work within the database. Then there is the question of inheritance. Within OSAM*, when an object inherits from another, it may inherit not only attributes, but also any rules and procedures such that the dependent object behaves in the same manner as its ancestor. In SQL, there is no such thing as inheritance. Columns defined in a given SQL table cannot be defined in terms of another column in another table. Each is its own entity; and, with the notable exception of declaring primary and foreign keys, do not interact with one another. Within the prototype, when an object inherits an attribute from another, this translates into a distinct SQL column being generated for the dependent object's corresponding table which bears the same name, length, and data type as its forbear. Certainly, if SQL were to be made fully compatible with the structures proposed within OSAM*, provisions for rule based logic and inheritance would have to be incorporated into SQL.
5.4 Other Issues

Of course, the prototype does bring up some questions not addressed by this study. There is little doubt that a similar prototype could be built to interface almost any semantic data model with SQL. But which is best? Surely, if the industry was to embrace the semantic data model as a design tool, a standard model would have to be decided upon. For the purposes of this study, OSAM* proved to be an excellent semantic data model on which to build the prototype. This is not to say that OSAM* would necessarily be the only candidate for a standard; but there are certain unique aspects of this model, such as its object-oriented approach, that might be desirable in a standard model. Perhaps the solution lies in a hybrid model which borrows from several prominent models in the literature. Such questions as these are left to further research.
REFERENCES


APPENDIX A

OSAM* Designer Code Listing

Application Description:
OSAM* Designer - This program enables a user who is familiar with the OSAM* Model to create and maintain an OSAM* s-diagram complete with regular entities, summary entities, and composite domains. When an s-diagram is completed, the program will take the OSAM* definition and generate SQL code to emulate the OSAM* structures. The program features a rudimentary graphical interface, and a complete set of File I/O utilities for creating and modifying OSAM* and SQL files. This program requires the SQLWindows runtime libraries available from Gupta Technologies, Inc. and Microsoft Windows 3.1 or higher.

Author: Paul F. Rabuck
Date: August 1, 1992

This software is submitted to the University of North Florida College of Computer and Information Sciences in partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences.

Outline Version - 3.0.A
Design-time Settings
Outline Window State: Maximized
Outline Window Location and Size
Left: 4.763"
Top: 0.031"
Width: 5.263"
Height: 5.094"
Options Box Location
Visible? No
Left: 4.8"
Top: 0.594"
Tool Palette Location
Visible? No
Left: 6.8"
Top: 2.25"

Included Objects

- winmenus.apl consists of a set of Windows functions for manipulating Windows menu objects.
  It includes the USER.EXE library.

- File Include: winmenus.apl

- winfiles.apl consists of a set of Windows functions for using the Windows 3.1 File Common Dialogs.
  It includes the SWCOMMON.DLL library.

- File Include: winfiles.apl

Global Declarations

Window Defaults
Form Window
  Font Name: System Default
  Font Size: System Default
  Font Enhancement: System Default
  Text Color: System Default
  Background Color: System Default

Dialog Box
  Font Name: MS Sans Serif
  Font Size: 8
  Font Enhancement: Bold
  Text Color: System Default
  Background Color: System Default
  Top Level Table Window
  Font Name: System Default
  Font Size: System Default
  Font Enhancement: System Default
Text Color: System Default
Background Color: System Default

Date Field
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Multiline Field
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Background Text
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Pushbutton
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent

Radio Button
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Check Box
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Group Box
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Child Table Window
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

List Box
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

ComboBox
Font Name: Use Parent
Font Size: Use Parent
Font Enhancement: Use Parent
Text Color: Use Parent
Background Color: Use Parent

Line
Line Color: Use Parent

Frame
Border Color: Use Parent
Background Color: Use Parent

Picture
Border Color: Use Parent
Background Color: Use Parent

Formats
Number: 0%
Number: #0
Number: #0.00
Number: $#0.00;(#$ #0.00)
Date/Time: hh:mm:ss AMPM
Date/Time: Mid/yy
Date/Time: MM-dd-yy
Date/Time: dd-MMM-yyyy
Date/Time: MMM d, yyyy hh:mm AMPM
Date/Time: MMMM d, yyyy hh:mm AMPM

External Functions
Constants
System
User
String: APPNAME = "OSAM* Designer"
I Windows System Commands
Number: WM_CHAR = 0x0102
I User Defined Messages
Number: MSG_Created = SAM_User + 1
Number: MSG_Gray = SAM_User + 2
Number: MSG_Hide = SAM_User + 3
Number: MSG_Show = SAM_User + 4
Number: MSG_DrawLine = SAM_User + 5
Number: MSG_Reset = SAM_User + 6
Number: MSG_Copy = SAM_User + 7
Number: MSG_Delete = SAM_User + 8
Number: MSG_Check = SAM_User + 9
Number: MSG_Change = SAM_User + 10
Number: MSG_Disable = SAM_User + 11
Number: MSG_HideLine = SAM_User + 12
Number: MSG_ShowLine = SAM_User + 13
Number: MSG_Redraw = SAM_User + 14
Number: MSG_Load = SAM_User + 15
Number: MSG_Highlight = SAM_User + 16
I OSAM Line Constants
Number: LN_8To10c = 61
Number: LN_7To9c = 60
Number: LN_2To4c = 59
Number: LN_1To3c = 58
Number: LN_1To7c = 57
Number: LN_4T010c = 56
Number: LN_7T010c = 55
Number: LN_1T03c = 54
Number: LN_1T02 = 53
Number: LN_1T03a = 52
Number: LN_1T03b = 51
Number: LN_1T04a = 50
Number: LN_1T04b = 49
Number: LN_1T05 = 48
Number: LN_1T06 = 47
Number: LN_1T07a = 46
Number: LN_1T07b = 45
Number: LN_1T06 = 44
Number: LN_1T08 = 43
Number: LN_1T010 = 42
Number: LN_2T03 = 41
Number: LN_2T04a = 40
Number: LN_2T04b = 39
Number: LN_2T05 = 38
Number: LN_2T06 = 37
Number: LN_2T07 = 36
Number: LN_2T08 = 35
Number: LN_2T09 = 34
Number: LN_2T10 = 33
Number: LN_3T04 = 32
Number: LN_3T05 = 31
Number: LN_3T06 = 30
Number: LN_3T07 = 29
Number: LN_3T08 = 28
Number: LN_3T09 = 27
Number: LN_3T010 = 26
Number: LN_4T05 = 25
Number: LN_4T06 = 24
Number: LN_4T07 = 23
Number: LN_4T08 = 22
Number: LN_4T09 = 21
Number: LN_4T010a = 20
Number: LN_4T010b = 19
Number: LN_5T06 = 18
Number: LN_5T07 = 17
Number: LN_5T08 = 16
Number: LN_5T09 = 15
Number: LN_5T010 = 14
Number: LN_6T07 = 13
Number: LN_6T08 = 12
Number: LN_6T09 = 11
Number: LN_6T010 = 10
Number: LN_7T08 = 9
Number: LN_7T09a = 8
Number: LN_7T09b = 7
Number: LN_7T010a = 6
Number: LN_7T010b = 5
Number: LN_8T09 = 4
Number: LN_8T010a = 3
Number: LN_8T010b = 2
Number: LN_9T010 = 1
Number: MAX_LINES = 62

1 MAX SCROLL must be an even number. MAX SCROLL determines the size of the
total workspace presented to the user. (ie. a MAX SCROLL of 6 will result in a
screen matrix with x and y coordinates ranging from -3 to 3 (G2)). Since the maximum
allowable lengths of dLnHBar and dLnVBar are preset to 2, these lengths may have
to be modified to accommodate MAX SCROLL if it is set above 18 (x and y ranges =
-9 to 9).
Number: MAX_SCROLL = 18

1
String: NULL = *
String: TAB = '    '
String: SPACE = '   '
String: DEL = '
Number: ALL = 0
Number: ENTITY-Regular = 1
Number: DOMAIN = 2
Number: ENTITY_CrossProduct = 3
Number: ENTITY_Composite = 4
Number: CUT = 1
Number: COPY = 2
Number: DELETE = 6
Number: DEPENDENT_UPON = 5
Number: DEPENDENT_ON_BY = 6
Number: ACTIVE = 0
String: CIRCLE_ON = 'ciyello.bmp'
String: CIRCLE_OFF = 'ciyelrn.bmp'
String: CIRCLE_CONNECTED = 'clrgreen.bmp'
String: SPACES = '   '
1 SQL Column Types
String: TYPE_CHOICES = 'CVLENUMRFLOAT'
Number: CHAR = 0
Number: VARCHAR = 1
Number: LONG = 2
Number: INTEGER = 3
Number: DECIMAL = 4
Number: NUMBER = 5
Number: SMALLINT = 6
Number: REAL = 7
Number: FLOAT = 8
Number: DOUBLE = 9
Number: DATE = 10
Number: TIME = 11
Number: TIMESTAMP = 12

Floating Menu Constants
Number: MENU_MAXLENGTH = 50
Number: SELECT_MENU = 3

String: REF_WARNING1 = 'The following statements will not work on SQL engines which do not support referential integrity.'
String: REF_WARNING2 = 'All tables referenced below and their corresponding unique indexes must be created prior to these statements being executed, or these statements will not work.'

Variables
  I Global Booleans
     I
     Boolean: bGloAddObject
     Boolean: bGloBuildSQL
     Boolean: bGloChanged
     Boolean: bGloDetailOK
     Boolean: bGloDuponCheck
     Boolean: bGloInitialized
     Boolean: bGloItemsVisible
     Boolean: bGloOverwrite
     Boolean: bGloReferential
     Boolean: bGloUpperCommands
     Boolean: bGloStoreComposites
     Boolean: bGloStoreCrossProducts
  I
  I Global Numbers
     I
     Number: nGloAbsPos
     Number: nGloCompNumber
     Number: nGloCount
     Number: nGloDetailObject
     Number: nGloEditPos
     Number: nGloErrorFound
     Number: nGloForeignCount
     Number: nGloNextPos
     Number: nGloObject
     Number: nGloRelPos
     Number: nGloScreen
     Number: nGloScreenPos
  I
  SQL File Array Variables
     I
     String: strGloRegularIndex[]
     String: strGloSQLTableDef[]
     String: strGloSQLForeignDef[]
     String: strGloStoredComposites[]
     String: strGloStoredCrossProducts[]
     String: strGloUniqueIndex
     String: strGloUniqueColumns
  I
  File Strings
     I
     String: strGloDefPath
     String: strGloFilePath
     String: strGloFileName
  I
  Miscellaneous Strings
     I
     String: strGloCSV
     String: strGloDupeAttr
     String: strGloErrorMessage
     String: strGloSQLReservedWord[]
  I
  I Screen "Cells"
  I
     Number: nGloCell[*]

- 70 -
Object Arrays

String: strGlo_ID[*]
String: strGloObjName[*]
String: strGloObjDesc[*]
Number: nGloObjType[*]
Number: nGloObjCell[*]
Number: nGloObjHBar[*]
Number: nGloObjVBar[*]
Number: nGloObjAttrPtr[*]
Long String: strGloObjAssociations[*]
Long String: strGloObjCRel[*]
Long String: strGloObjXRel[*]
Long String: strGloObjInsert[*]
Long String: strGloObjUpdate[*]
Long String: strGloObjDelete[*]

Related Objects Arrays

Number: nGloRelated[*]
Number: nGloRelatedType[*]
Number: nGloRelatedLevel[*]
String: strGloObjRelation[*]

Detail Item Arrays

String: strGloAttrName[*]
Boolean: bGloAttrKey[*]
Boolean: bGloAttrRequired[*]
Boolean: bGloAttrIndexed[*]
Number: nGloAttrType[*]
Number: nGloAttrLength[*]
Number: nGloAttrScale[*]
Number: nGloAttrStatus[*]
Number: nGloAttrNextPtr[*]
String: strGloAttrComments[*]
String: strGloAttrCRel[*]
String: strGloAttrXRel[*]

Detail Item Copy

String: strGloAttrCopyName[*]
Boolean: bGloAttrCopyKey[*]
Boolean: bGloAttrCopyRequired[*]
Boolean: bGloAttrCopyIndexed[*]
Number: nGloAttrCopyType[*]
Number: nGloAttrCopyLength[*]
Number: nGloAttrCopyScale[*]
String: strGloAttrCopyComments[*]
String: strGloAttrCopyCRel[*]
String: strGloAttrCopyXRel[*]
Long String: strGloAttrCopyRules[*]

Global Window Handles

Window Handle: hWndGloDetailTable
Window Handle: hWndGloMenuBar
Window Handle: hWndGloSubMenu
Window Handle: hWndGloWait

included Strings (from winfiles.spi)

Internal Functions

Function: AddAssociations
Description: Creates an association between two objects. The associations for any object are stored in the global strGloObjAssociations[*] array as a CSV. This function updates the strGloObjAssociation array and passes a message to the workspace to redraw itself.

Returns
Parameters
Number: nAddObj
Local variables
String: strAddObj
String: strAddObjID
Number: nObjLoc
String: strObjLoc
String: strInverseAssoc

Actions
Call SaINumberToStr(nAddObj,0,strAddObj)
Set strGloCSV = strGloObjAssociations[nAddObj]
While SaINStrLength(strGloCSV) > 1
  Set strObjectID = ReadCSV()
  Call SaIStrRight(strObjectID,1,strInverseAssoc)
  If strInverseAssoc > 'Z'
    Call SaIStrUpper(strInverseAssoc,strInverseAssoc)
  Else
    Call SaIStrLower(strInverseAssoc,strInverseAssoc)
  Call SaIStrLeft(strObjectID,SaIStrScan(strObjectID,','),strObjLoc)
  Call SaIStrLop(strObjLoc)
  Set nObjLoc = SaIStrToNumber(strObjLoc)
  Set strGloObjAssociations[nObjLoc] = strGloObjAssociations[nObjLoc] || '#'||strAddObj||'::'||strInverseAssoc||'
Call SaIPostMsg(frmObjMgr.pbOrigin,MSG_Redraw,O,O)

Function: AppendSettingsToType
Description: Retrieves the SQL length and scale parameters for a given attribute and produces a more readable display of the attribute type in the case of CHARs, VARCHARs or DECIMALs.
Returns
Parameters
Local variables
String: strLength

Actions
If colType = 'Character Field' or colType = 'Variable Length Field'
  Call SaINumberToStr(nLength,0,strLength)
  If colType = 'Character Field'
    Set colType = strLength||SPACE||colType
  Else
    Set colType = strLength||' char.'||colType
  Set colScale = -1
If colType = 'Decimal'
  Set strLength = NULL
  While nLength > 0
    If strLength = '999' or strLength = '999,999' or strLength = '999,999,999'
      Set strLength = ',||strLength
    Set strLength = '||strLength
    Set nLength = nLength -1
  If nScale > 0
    If strLength = NULL
      Set strLength = '0.'
    Else
      Set strLength = strLength||'
      While nScale > 0
        Set strLength = strLength||'
      Set nScale = nScale -1
    Set colType = colType||('||strLength))'

Function: BuildSQLFile
Description: Coordinates the creation of a SQL file after the user has selected the desired options on the Build SQL Dialog box, and has entered a file name.
Returns
Parameters
String: strFileName
Window Handle: hWndList

Local variables
File Handle: hFile
Number: nMaxCount
Number: nColCount
Number: nListCount
Number: nLine
String: strLine
String: strObjParm[5]
String: strObject
String: strOverwrite
String: strTable
String: strErrors

Actions
Set nGlobalErrorFound = 0
If SaFileOpen(hFile,strFileName,OF_Create|OF_Write)
    Call SaWaitCursor(TRUE)
    Set bGlobalBuildSQL = TRUE
    Set strMessage = 'Building SQL File: ['strFileName]' ...
If cbInclude
    Call SaFilePutStr(hFile,dfDelimiter|SPACE|APPNAME)
    Call SaFilePutStr(hFile,dfDelimiter|"Copyright @ 1992 by Paul F. Rabuck")
    Call SaFilePutStr(hFile,dfDelimiter)
    Call SaFilePutStr(hFile,NULL)
If cbRules
    If rbOverwrite
        If bGlobalUpperCommands
            Call SaStrLower(dfRulesTable,strGloSQLTableDef[0])
            Set strGloSQLTableDef[0] = 'DROP TABLE ['strGloSQLTableDef[0]];'
        Else
            Set strGloSQLTableDef[0] = 'DROP table ['dfRulesTable];'
            Set strGloSQLTableDef[1] = NULL
        Call WriteSQLToFile(hFile,strGloSQLTableDef[0],rbProper,rbUpper,rbLower,FALSE,FALSE)
    If bGlobalUpperCommands
        Call SaStrLower(dfRulesTable,strGloSQLTableDef[0])
        Set strGloSQLTableDef[0] = 'CREATE TABLE ['strGloSQLTableDef[0]];'
        Set strGloSQLTableDef[1] = 'bname CHAR(18) NOT NULL,'
        Set strGloSQLTableDef[2] = 'on_insert CHAR(254),'
        Set strGloSQLTableDef[3] = 'on_update CHAR(254),'
        Set strGloSQLTableDef[4] = 'on_delete CHAR(254),'
    Else
        Set strGloSQLTableDef[0] = 'CREATE table ['dfRulesTable];'
        Set strGloSQLTableDef[1] = 'bname char(18) not null,'
        Set strGloSQLTableDef[2] = 'ON_INSERT char(254),'
        Set strGloSQLTableDef[3] = 'ON_UPDATE char(254),'
        Set strGloSQLTableDef[4] = 'ON_DELETE char(254),'
    Set strGloSQLTableDef[5] = NULL
    Call WriteSQLToFile(hFile,strGloSQLTableDef[0],rbProper,rbUpper,rbLower,FALSE,FALSE)
Set nListCount = 0
Set nGlobalForeignCount = 0
Set nMaxCount = SaIstQueryCount(hWndList)
While nListCount < nMaxCount
    If not cbRestrict or SaIstQueryState(hWndList,nListCount)
        Call SaIstQueryText(hWndList,nListCount,strObject)
        Call SaStrTokenize(strObject,TAB,TAB,strObjParm)
        Set nGlobalDetailObject = SaStrToNumber(strObjParm[1])
    If cbInclude
        Call SaFilePutStr(hFile,dfDelimiter|" SQL Definition for: ['strGloObjName[nGlobalDetailObject]]")
        Call SaFilePutStr(hFile,dfDelimiter|"")
        Call WordWrap(hFile,strGloObjDesc[nGlobalDetailObject],dfDelimiter,NOTES: ',FALSE)
        Call WordWrap(hFile,strGloObjInsert[nGlobalDetailObject],dfDelimiter,ON_INSERT: ',FALSE)
        Call WordWrap(hFile,strGloObjUpdate[nGlobalDetailObject],dfDelimiter,ON_UPDATE: ',FALSE)
        Call WordWrap(hFile,strGloObjDelete[nGlobalDetailObject],dfDelimiter,ON_DELETE: ',FALSE)
    Call SaFilePutStr(hFile,dfDelimiter)
    Call SaFilePutStr(hFile,NULL)
If cbRules and
    (strGloObjDesc[nGlobalDetailObject] != NULL or
    strGloObjInsert[nGlobalDetailObject] != NULL or
    strGloObjUpdate[nGlobalDetailObject] != NULL or
    strGloObjDelete[nGlobalDetailObject] != NULL)
If bGlobalUpperCommands
    Call SaStrLower(dfRulesTable,strLine)
    Set strLine = 'INSERT INTO ['strLine]|VALUES'
Else
    Set strLine = 'insert into ['dfRulesTable]|values'
If rbProper
    Call SaStrProper(strLine,strLine)
If rbUpper
  Call SaIStrUpper(strLine, strLine)
If rbLower
  Call SaIStrLower(strLine, strLine)
Call SaIFilePutStr(hFile, strLine)
Call WordWrap(hFile, [strGloObjName[nGloDetailObject]]; NULL, NULL, FALSE)
Call WordWrap(hFile, [strGloObjInsert[nGloDetailObject]]; NULL, NULL, FALSE)
Call WordWrap(hFile, [strGloObjUpdate[nGloDetailObject]]; NULL, NULL, FALSE)
Call WordWrap(hFile, [strGloObjDelete[nGloDetailObject]]; NULL, NULL, FALSE)
Call SaIFilePutStr(hFile, NULL)
If rbOverwrite
  If bgloUpperCommands
    Call SaIStrLower(strGloObjName[nGloDetailObject], strOverwrite)
    Set strOverwrite = 'DROP TABLE ' + strOverwrite;
  Else
    Set strOverwrite = 'drop table ' + strGloObjName[nGloDetailObject]; ;
  EndIf
  If rbProper
    Call SaIStrProper(strOverwrite, strOverwrite)
  EndIf
  If rbUpper
    Call SaIStrUpper(strOverwrite, strOverwrite)
  EndIf
  If rbLower
    Call SaIStrLower(strOverwrite, strOverwrite)
    Call SaIFilePutStr(hFile, strOverwrite)
    Call SaIFilePutStr(hFile, NULL)
  EndIf
EndIf
EndIf
EndIf
EndIf
Else
  If strGloSQL TableDef[0] = NULL
    Call WordWrap(hFile, 'No Attributes Exist For ' + strGloObjName[nGloDetailObject]]; Table ' + strGloObjName[nGloDetailObject]; could not be created.';
    dfDelimiter, ERROR: ', FALSE)
    Set nGloErrorFound = nGloErrorFound + 1
    Call SaIFilePutStr(hFile, dfDelimiter)
    Call SaIFilePutStr(hFile, NULL)
  Else
    Call WriteSQLToTable(hFile, strGloSQL TableDef, rbProper, rbUpper, rbLower, FALSE, TRUE)
    If rbProper
      Call SaIFilePutStr(hFile, dfDelimiter)
    EndIf
    If rbUpper
      Call SaIStrUpper(strGloUniqueIndex, strGloUniqueIndex)
    EndIf
    If rbLower
      Call SaIStrLower(strGloUniqueIndex, strGloUniqueIndex)
      Call WordWrap(hFile, strGloUniqueIndex, NULL, NULL, FALSE)
    EndIf
    If strGloUniqueIndex != NULL
      Call SaIFilePutStr(hFile, NULL)
      Set dfGloUniqueIndex = NULL
      Call WriteSQLToTable(hFile, strGloRegularIndex, strGloRegularIndex, NULL, NULL, TRUE)
      Call WriteSQLToTable(hFile, strGloStoredCrossProducts, strGloStoredCrossProducts, TRUE)
      If cbStoreCrossProducts
        Call WriteSQLToTable(hFile, strGloStoredCrossProducts, TRUE)
      EndIf
      If cbStoreComposites
        Call WriteSQLToTable(hFile, strGloComposites, TRUE)
      EndIf
    EndIf
EndIf
EndIf
EndIf
Set nListCount = nListCount + 1
If nGloForeignCount > 0
  If cbInclude
    Call SaIFilePutStr(hFile, dfDelimiter; SQL Foreign Key Definitions)
    Call SaIFilePutStr(hFile, dfDelimiter; SQL Foreign Key Definitions)
    Call WordWrap(hFile, dfDelimiter, 'WARNING 1', FALSE)
    Call WordWrap(hFile, dfDelimiter, 'WARNING 2', FALSE)
    Call SaIFilePutStr(hFile, dfDelimiter)
  EndIf
  Call SaIFilePutStr(hFile, NULL)
EndIf
Call WriteSQLToTable(hFile, strGloSQLForeignDef, rbProper, rbUpper, rbLower, TRUE)
If cbCommit
  If rbCommands or rbUpper
    Call SaIFilePutStr(hFile, COMMIT;)
  EndIf
  If rbObjects or rbLower
    Call SaIFilePutStr(hFile, commit;)
  EndIf
  If rbProper
    Call SaIFilePutStr(hFile, 'Commit;')
  Call SaIFileClose(hFile)
EndIf
Set strMessage = NULL
- 74 -
Set bGloBuildSQL = FALSE
If nGloErrorFound > 0
    Call SalNumberToStr(nGloErrorFound, 0, strErrors)
    Call SalMessageBox(strErrors & " Error(s) were found...please check the SQL file for errors.", APPNAME, MB_OK | MB_ICONEXCLAMATION)
    Call SalLoadApp("notepad", strFileName)
    Call SalWaitCursor(FALSE)
End If

Function: CopyDetails
Description: Copies the attributes for a given object to temporary variables for later retrieval (used when doing an edit/copy, or when the program checks for duplicates.)
Returns
Boolean:
Parameters
Number: nObjectID
Boolean: bTempCopy
Local variables
Number: nCopyCount
Actions
If nGloObjectPtr(nObjectID) = 0
    Set strGloAttrCopyName[0] = NULL
    Return TRUE
Else
    Set nCopyCount = 0
    Set nNextPtr = nGloObjectPtr[nObjectID]
    While nNextPtr 1=-1
        Set strGloAttrCopyName[nCopyCount] = strGloAllrName[nNextPtr]
        Set bGloAttrCopyKey[nCopyCount] = bGloAttrKey[nNextPtr]
        Set bGloAttrCopyRequired[nCopyCount] = bGloAttrRequired[nNextPtr]
        Set bGloAttrCopyIndexed[nCopyCount] = bGloAttrIndexed[nNextPtr]
        Set nGloAttrCopyLength[nCopyCount] = nGloAttrLength[nNextPtr]
        Set nGloAttrCopyScale[nCopyCount] = nGloAttrScale[nNextPtr]
        Set strGloAttrCopyComments[nCopyCount] = strGloAttrComments[nNextPtr]
        If bTempCopy
            Set strGloAttrCopyCReI[nCopyCount] = strGloAttrCReI[nNextPtr]
            Set strGloAttrCopyXReI[nCopyCount] = strGloAttrXReI[nNextPtr]
        End If
        Set nCopyCount = nCopyCount + 1
        Set nNextPtr = nGloObjectPtr[nNextPtr]
    End While
    Return TRUE
End If

Function: CopyObject
Description: Copies the definition and associations for a given object to temporary variables for later retrieval (used when doing an edit/copy.)
Returns
Parameters
Number: nCopyFrom
Number: nCopyTo
Local variables
Actions
Set strGlo_ID[nCopyTo] = strGlo_ID[nCopyFrom]
Set strGlo_Name[nCopyTo] = strGlo_Name[nCopyFrom]
Set strGlo_Description[nCopyTo] = strGlo_Description[nCopyFrom]
Set nGloObjectType[nCopyTo] = nGloObjectType[nCopyFrom]
Set nGloObjectCell[nCopyTo] = nGloObjectCell[nCopyFrom]
Set nGloObjectHalfBar[nCopyTo] = nGloObjectHalfBar[nCopyFrom]
Set nGloObjectBar[nCopyTo] = nGloObjectBar[nCopyFrom]
Set nGloObjectAttrPtr[nCopyTo] = nGloObjectAttrPtr[nCopyFrom]
Set strGloObjectAssociations[nCopyTo] = strGloObjectAssociations[nCopyFrom]
Set strGloObjectInsert[nCopyTo] = strGloObjectInsert[nCopyFrom]
Set strGloObjectUpdate[nCopyTo] = strGloObjectUpdate[nCopyFrom]
Set strGloObjectDelete[nCopyTo] = strGloObjectDelete[nCopyFrom]
If nCopyTo > 0
    Call AddAssociations(nCopyTo)
End If

Function: CreateIndexes
Description: Generates the SOL "Create Index..." statements for a given object and stores them to a temporary array. Called from the BuildSOLFile() function through the Details dialog.
Returns
Boolean:
Parameters
Local variables
String: strIndexName
String: strTableName
Number: nRegCount
String: strRegCount
Actions
Set strGloUniqueIndex = NULL
Set strGloUniqueColumns = NULL
Set nAttributeTblRow = TBL_MinRow
Set nRegCount = 0
Call SalStrToLower(strGloObjName[nGloDetailObject], strIndexName)
Call SalStrToLower(strGloObjName[nGloDetailObject], strTableName)
While SalTblFindNextRow(hWndForm, nAttributeTblRow, 0, 0)
Call SalTblSetContext(hWndForm, nAttributeTblRow)
If collKeyed = 'Yes'
If collKey = 'Yes'
If strGloUniqueIndex != NULL
Set strGloUniqueIndex = strGloUniqueIndex,"I'
Set strGloUniqueIndex = strGloUniqueIndex[coName]
Else
Call SalNumberToStr(nRegCount+1,0,strRegCount)
Set strGloRegularIndex[nRegCount] = 'CREATE INDEX"[strIndex].Name[",Index",strRegCount]" ON "[strTableName][["coName"]]]';
Set nRegCount = nRegCount + 1
Set strGloRegularIndex[nRegCount] = NULL
If strGloUniqueIndex != NULL
Set strGloUniqueColumns = strGloUniqueIndex
Set strGloUniqueIndex = 'CREATE UNIQUE INDEX [strIndex].Name[",key" ON "[strTableName][["coName"]]]';
Return TRUE
Function: CreateStoredSQL
Description: Generates the SQL "Store...Select" statements for a given object (to represent summary objects) and stores them to a temporary array. Called from the BuildSQLFile() function through the Details dialog.
Returns
Boolean:
Parameters
Number: nEntityType
Local variables
String: strComposition
String: strTable
String: strTableName
String: strName
String: strCompDef
String: strCompNumber
Number: nDetailTblRow
Number: nCompCount
Actions
Set nGloCompNumber = 0
Set nCompCount = 0
While nGloRelated[nGloCompNumber] != -1
If nGloRelatedType[nGloCompNumber] = nEntityType
Call SalNumberToStr(nGloRelated[nGloCompNumber], 0, strCompNumber)
If nEntityType = ENTITY_Composite
Set strCompDef = strGloObjRel[nGloDetailObject]
Set strTable = strGloID[nGloRelated[nGloCompNumber]]["_oF_"[strGloID[nGloDetailObject]]][]
Else
Set strCompDef = strGloObjRel[nGloDetailObject]
Set strTable = strGloID[nGloDetailObject]["_by_"[strGloID[nGloRelated[nGloCompNumber]]][]
If SalStrScan(strCompDef, strCompNumber) != -1
Set strComposition = 'COUNT(*)'
Set strTable = strTable["strGloID[nGloDetailObject]"_count,']
Set nDetailTblRow = TBL_MinRow
While SalTblFindNextRow(hWndGloDetailTable, nDetailTblRow, 0, 0)
Call SalTblSetContext(hWndGloDetailTable, nDetailTblRow)
If colItemType > 2 and collItemType < 10
Set strName = hWndGloDetailTable.colName
If nEntityType = ENTITY_Composite
Set strCompDef = hWndGloDetailTable.colCRel
Else
    Set strCompDef = hWndGloDetailTable.colXRel
If SaIStrScan(strCompDef,'#')I=-1
    Call SalStrMid(strCompDef,SaIStrScan(strCompDef,'#')I,strLength(strCompDef)+7,strCompDef)
Else
    Set strCompDef = NULL
If SaIStrScan(strCompDef,'L')I=-1
    Set strComposition = strComposition: 'MIN('IstrNameI')','
    Set strTable = strTable: 'N','
If SaIStrScan(strCompDef,'F')I=-1
    Set strComposition = strComposition: 'MAX('IstrNameI')','
    Set strTable = strTable: 'X','
If SaIStrScan(strCompDef,'A')I=-1
    Set strComposition = strComposition: 'AVG('IstrNameI')','
    Set strTable = strTable: 'A','
If SaIStrScan(strCompDef,'T')I=-1
    Set strComposition = strComposition: 'SUM('IstrNameI')','
    Set strTable = strTable: 'S','
If strComposition I= NULL
    Call SaIStrLeft(strComposition,SaIStrLength(strComposition)-2,strComposition)
    Call SaIStrLower(strTable,strTable)
    Call SaIStrLower(strObjName(nGIoDetailObject),strTableName)
If nEntityType = ENTITY_Composite
    If bGloOverwrite
        Set strGloStoredComposites[nCompCount] = 'ERASE ','
    Set strGloStoredCrossProducts[nCompCount] = SPACE
    Set nCompCount = nCompCount + 2
    Set strComposition =
        'CREATE VIEW ','AS SELECT ',' FROM ','
    Else
        If bGloOverwrite
            Set strGloStoredCrossProducts[nCompCount] = 'ERASE ','
    Set strGloStoredCrossProducts[nCompCount+1] = SPACE
    Set nCompCount = nCompCount + 2
    Set strComposition =
        'CREATE VIEW ','AS SELECT ',' FROM ','
    If nEntityType = ENTITY_Composite
        Set strGloStoredComposites[nCompCount] = strComposition
    Else
        Set strGloStoredCrossProducts[nCompCount] = strComposition
        Set nCompCount = nCompCount + 1
    If nEntityType = ENTITY_Composite
        Set strGloStoredCrossProducts[nCompCount] = NULL
    Else
        Set strGloStoredCrossProducts[nCompCount] = NULL
    Return TRUE
Function: CreateTable
Description: Generates the SQL "Create Table..." and "Comment On..." statements for a given object
and stores them to a temporary array. Called from the BuildSQLFile() function through
the Details dialog.
Returns
Boolean:
Parameters
Number: nTable
Local variables
String: strSQL
String: strTable
Number: nColCount
String: strType
String: strName
String: strLength
String: strScale
String: strComment[ ]
Actions
Set nAttributeTblRow = TBL_MinRow
Set strTable = strGloObjName[nTable]
Set nColCount = 1
While SaITblFindNextRow(hWndForm,nAttributeTblRow,0,0)
    Call SaITblSetContext(hWndForm,nAttributeTblRow)
    If strSQL = NULL
        Set strSQL = strSQL[['']
        Set strGloSQLTableDef[nColCount] = strSQL
        Set strSQL = NULL
        Set nColCount = nColCount + 1
    Call SaIStrLeft(co1Name,SPACES,20,strName)
    If not bGloUpperCommands
        Call SaIStrUpper(strName,strName)
        Set strSQL = strSQL[['strName']
    Select Case collernType
        Case CHAR
            Call SaINumberToStr(co1Length,0,strLength)
            Set strType = 'CHAR('''strLength''')'
            Break
        Case VARCHAR
            Call SaINumberToStr(co1Length,0,strLength)
            Set strType = 'VARCHAR('''strLength''')'
            Break
        Case LONG
            Set strType = 'LONG'
            Break
        Case INTEGER
            Set strType = 'INTEGER'
            Break
        Case DECIMAL
            Call SaINumberToStr(co1Length,0,strLength)
            Call Sa1NumberToStr(co1Scale,0,strScale)
            Set strType = 'DECIMAL('''strLength''',''strScale''')'
            Break
        Case NUMBER
            Set strType = 'NUMBER'
            Break
        Case SMALLINT
            Set strType = 'SMALLINT'
            Break
        Case REAL
            Set strType = 'REAL'
            Break
        Case FLOAT
            Set strType = 'FLOAT'
            Break
        Case DOUBLE
            Set strType = 'DOUBLE'
            Break
        Case DATE
            Set strType = 'DATE'
            Break
        Case TIME
            Set strType = 'TIME'
            Break
        Case TIMESTAMP
            Set strType = 'TIMESTAMP'
            Break
    If not bGloUpperCommands
        Call SaIStrLower(strType,strType)
        Set strSQL = strSQL[['strType']
    If colRequired = 'Yes'
        If bGloUpperCommands
            Set strSQL = strSQL[['NOT NULL'']
        Else
            Set strSQL = strSQL[[' not null']
    If strSQL = NULL
        If bGloUpperCommands
            Set strSQL = strSQL[['']
        Else
            Set strSQL = strSQL[[''] not null
        If strSQL = NULL
        If bGloUpperCommands

- 78 -
Call SaisStrLower(strTable,strTable)
Set strGloSQLTableDef[0] = 'CREATE TABLE ['||strTable||']'
Else
    Call SaisStrUpper(strTable,strTable)
    Set strGloSQLTableDef[0] = 'CREATE TABLE ['||strTable||']'
If strGloUniqueColumns = NULL or not cbReferential
    Set strGloSQLTableDef[nColCount] = strSQL;
Else
    Set strSQLGloSQLTableDef[nColCount] = strSQL;
    Set nColCount = nColCount + 1
    If bGloUpperCommands
        Call SaisStrLower(strGloUniqueColumns,strGloUniqueColumns)
        Set strSQLGloSQLTableDef[nColCount] = 'PRIMARY KEY ('||strGloUniqueColumns||')';
    Else
        Call SaisStrUpper(strGloUniqueColumns,strGloUniqueColumns)
        Set strSQLGloSQLTableDef[nColCount] = 'primary key ('||strGloUniqueColumns||')';
        Set strSQLGloSQLTableDef[nColCount] = strSQL;
        Set nColCount = nColCount + 1
        If not SaITblFirst(hWndForm,nAttributeTblRow)
            While SaITblFindNextRow(hWndForm,nAttributeTblRow,0,0)
                Call SaisStrContext(hWndForm,nAttributeTblRow)
                If not SaisNull(coIComments)
                    If bGloUpperCommands
                        Call SaisStrLower(coIName,strName)
                        Set strSQLGloSQLTableDef[nColCount] = 'COMMENT ON COLUMN ['||strTable||'].['||strName||'] IS '''||coIComments||'''
                    Else
                        Call SaisStrUpper(coIName,strName)
                        Set strSQLGloSQLTableDef[nColCount] = 'comment on column ['||strTable||'].['||strName||'] IS '''||coIComments||'''
                    Set nColCount = nColCount + 1
                    If bGloReferential
                        Call CreateForeignKeys(strTable)
                    Else
                        Set strSQLGloSQLTableDef[0] = NULL
                    Return TRUE
        Else
            Set strSQLGloSQLForeignDef[nColCount] = strSQL
        Return TRUE
Function: CreateForeignKeys
Description: Generates the SQL "ALTER TABLE...FOREIGN KEY" statements for a given object when the
user selects referential integrity in the Build SQL options and stores it to a temporary array.
Called from the BuildSQLFile() function through the Details dialog.

Returns
    Boolean:
Parameters
    String: strTable
Local variables
    String: strSQL
    String: strName
    String: coIFrom
    String: strAssocType
    Number: nColCount
    String: strTableName
Actions
    Set strSQL = NULL
    Set nColCount = nGloForeignCount + 1
    Set nAttributeTblRow = TBL_MinRow
    While SaITblFirst(hWndForm,nAttributeTblRow,0,0)
        Call SaisStrContext(hWndForm,nAttributeTblRow)
        If coIFrom = NULL and coIAssocType = 'G'
            If strSQL = NULL
                Set strSQL = strSQL' '',
                Set strSQLGloSQLForeignDef[nColCount] = strSQL
Set strSQL = NULL
Set nColCount = nColCount + 1
Set strFrom = colFrom
Set strTableName = colTableName
Set strAssocType = colAssocType
Call SaIStrLower(coIFrom,coIFrom)
Set strSQL = 'FOREIGN KEY [licolFrom] ll REFERENCES [' + strTableName + ']' ON DELETE SET NULL
While colFrom = strFrom and colFrom = NULL
Call SaIStrLower(colName,coINamel)
Set strSQL = strSQL + coINamel + ':
If SaITbIFlndNextRow(hWndForm,nAttributeTbIRow,O,O)
Call SaITbISetContext(hWndForm,nAttributeTbIRow)
Else
Set coIFrom = NULL
Call SaIStrLeft(strSQL,SaIStrLength(strSQL)-1,strSQL)
Set nAttributeTblRow = nAttributeTblRow - 1
Call SaIStrLower(strTableName,strTableName)
If strAssocType = 'A'
Set strSQL = strSQL + ' REFERENCES [' + strTableName + '] ON DELETE SET NULL'
Else
Set strSQL = strSQL + ' REFERENCES [' + strTableName + '] ON DELETE RESTRICT'
If nColCount > nGloForeignCount+1
Set strSQLGloSQLForeignDef[nColCount] = strSQL + ';
Call SaIStrLower(strTableName,strTableName)
Set strSQLGloSQLForeignDef[nGloForeignCount] = 'ALTER TABLE ' + strTableName
Set strSQLGloSQLForeignDef[nColCount+1] = SPACE
Set strSQLGloSQLForeignDef[nColCount+2] = NULL
Set nGloForeignCount = nColCount + 2
Return TRUE

Function: Delete Associations
Description: Removes an association between two objects. The associations for any object are stored in the global strGloObjAssociations[] array as a CSV. This function updates the strGloObjAssociation array and posts a message to the workspace to refresh itself.

Returns
Parameters
Number: nDelObj
Boolean: bRefresh
Local variables
String: strDelObj
String: strObjectID
Number: nObjLoc
String: strObjLoc
Actions
Call SaINumberToStr(nDelObj,O,strDelObj)
Set strGloCSV = strGloObjAssociations[nDelObj]
While SaIStrLength(strGloCSV) > 1
Set strObjectID = ReadCSV()
Call SaIStrLeft(strObjectID,SaIStrScan(strObjectID::'),strObjLoc)
Call SaIStrReplace(strObjectID,strObjLoc)
Set nObjLoc = SaIStrToNumber(strObjLoc)
Call SaIStrReplace(strGloObjAssociations[nObjLoc],
SaIStrScan(strGloObjAssociations[nObjLoc],',')
Set strGloObjAssociations[nObjLoc]+4,NULL,strGloObjAssociations[nObjLoc])
Set strGloObjAssociations[nDelObj] = NULL
If bRefresh
Call RefreshLabels()

Function: Delete Object
Description: Removes the definition and associations for a given object and posts a message to the workspace to redraw itself. Performed on an edit/cut, or an edit/clear.

Returns
Parameters
Number: nDelete
Local variables
Actions
Call DeleteAssociations(nDelete,TRUE)
Set strGloO_ID[nDelete] = NULL
Set strGloObjName[nDelete] = NULL
Set strGloObjDesc[nDelete] = NULL
Set nGloObjType[nDelete] = DELETE
Set nGloObjCell[nDelete] = 0
Set nGloObjHBar[nDelete] = -1
Set nGloObjVBar[nDelete] = -1
Set nGloObjAttrPtr[nDelete] = 0
Set nGloCell[nGloAbsPos] = 0
Set strGloObjInsert[nDelete] = NULL
Set strGloObjUpdate[nDelete] = NULL
Set strGloObjDelete[nDelete] = NULL
Call SaISendMsgToChildren(hWndForm.MSG_Disable,0,0)
Set wParam = SaINumberMod(nGloAbsPos,10)
If wParam = 0
  Set wParam = 10
Call SaISendMsgToChildren(hWndForm.MSG_HideLine,wParam,0)
Call SaISendMsg(hWndDF[nGloScreenPos].MSG_Delete,0,0)
Call SaISendMsg(hWndDF[nGloScreenPos].MSG_Hide,0,0)
Call Redraw()

Function: FileContinue
Description: Prompts the user to save any changes made to their s-diagram before continuing. Performed on file/new or file/open.
Returns
Boolean:
Parameters
Local variables
Number: nSaveResponse
String: strCapFile
Actions
If bGloChanged
  If strGloFileName = NULL
    Set nSaveResponse = SaIMessageBox("Save Changes To File?, APPNAME_MB_YesNoCancel\MB IconQuestion")
  Else
    Call SaIStrUpper(strGloFileName,strCapFile)
    Set nSaveResponse = SaIMessageBox("Save Changes To \"strCapFile\"?, APPNAME_MB_YesNoCancel\MB IconQuestion")
  If nSaveResponse = IDYES
    Call ResetFileNames(strGloFileName,strGloFilePath,'sam')
    If strGloFilePath = NULL
      Call SaIStrLen(strGloFileName,SaIStrScan(strGloFileName,'.').strGloFileName)
      Set strGloFileName = strGloFileName'.sam.
    If DlgSaveAs(hWndForm,strGloDefPath.APPNAMEII- -Save File',\'sam',OSAM Files-.sam-.OFN_PATHMUSTEXIST|OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT,strGloFilePath,strGloFileName)
      Call SaveOSAMFile(strGloFilePath)
    Else
      Return FALSE
  Else
    Call SaveOSAMFile(strGloFilePath)
    If nSaveResponse = IDCANCEL
      Return FALSE
  Return TRUE

Function: FileNew
Description: Initializes all workspace and attribute arrays and posts a message to pbOrigin to return the workspace to screen (0,0). Performed on file/new when the user wishes to start a brand new s-diagram.
Returns
Boolean:
Parameters
Boolean: bRedraw
Local variables
Number: nDeleteCell
Number: nMaxCell
Number: nDeleteObject
Actions
If not bGloInitialized or bGloChanged
  If FileContinue()
    Call SaIWailCursor(TRUE)
    Set strMessage = "Initializing Workspace...
    Set nGloObjFunction = DELETE
    - 81 -
Set nDelObject = 1
While nGloObjType[nDelObject] = 0
   Call DeleteAssociations(nDelObject, FALSE)
   Set strGloObjID[nDelObject] = NULL
   Set strGloObjName[nDelObject] = NULL
   Set strGloObjDesc[nDelObject] = NULL
   Set nGloObjType[nDelObject] = DELETE
   Set nGloObjCell[nDelObject] = 0
   Set nGloObjAttrPtr[nDelObject] = 0
   Set strGloObjInsert[nDelObject] = NULL
   Set strGloObjUpdate[nDelObject] = NULL
   Set strGloObjDelete[nDelObject] = NULL
   Set nDelObject = nDelObject + 1
Set nDeleteCell = 0
Set nMaxCell = (SaINumberPower(MAX_SCROLL+1,2)+MAX_SCROLL)*10
While nDeleteCell < nMaxCell
   Set nGloCell[nDeleteCell] = 0
   Set nDeleteCell = nDeleteCell + 1
   Set bEntityFocus = FALSE
   Set bGloChanged = FALSE
   Set bGloInitialized = TRUE
   Call SaWaitCursor(FALSE)
   If bRedraw
      Call SaSetWindowText(hWndForm, APPNAMEII' - (untitled)'
      Call Redraw()}
      Call SaSendMsgToChildren(hWndForm, MSG_Disable, 0, 0)
      Set bStartApp = TRUE
      Call SaSendMsg(pbOrigin, SAM_Click, 0, 0)
      Return TRUE
   Else
      Return FALSE
   Else
      Return TRUE
Function: FlipCase
Description: Called depending on the Text Format options select on the Build SQL Dialog, called by the WordWrap() and WriteSOLToFile() functions.
Returns
String:
Parameters
String: strText
Local variables
Number: nWordCount
String: strWord[*]
String: strWordCheck
Number: nTokens
Actions
Set nTokens = SaStrTokenize(strText, SPACE, SPACE, strWord)
Set strText = NULL
Set nWordCount = 0
While nWordCount < nTokens
   Call SaStrUpper(strWord[nWordCount], strWordCheck)
   If strWordCheck = strWord[nWordCount]
      Call SaStrLower(strWordCheck, strWordCheck)
   Set strText = strText || SPACE || strWordCheck
   Set nWordCount = nWordCount + 1
Call SaStrLop(strText)
Return strText
Function: GetAllRelatedObjects
Description: Lists data concerning either ancestor or dependent objects for a given object into four separate receive arrays depending upon the bDependent boolean (If FALSE, ancestors; If TRUE, dependents) regardless of how many levels the dependency is removed from the given object.
Returns
Parameters
Number: nObject
Boolean: bDependent
Receive Number: nRelated[*]
Receive Number: nRelatedLevel[*]
Receive Number: nRelatedType[*]
Receive String: strObjRelation[*]

Local variables
Number: nStart
Number: nLevelStart
String: strBaseRelation

Actions
Set nRelated[0] = nObject
Set nRelatedLevel[0] = 0
Set nRelated[1] = -1
Set nGloCount = 0
Set nStart = 1
Set nLevelStart = 1
While nRelated[nGloCount] != -1
  Set nStart = GetDirectlyRelatedObjects(nRelated[nGloCount], nStart, bDependent, strObjRelation[nGloCount], nRelated[nRelatedType], strObjRelation)
While nLevelStart != nStart
  Set nRelatedLevel[nLevelStart] = nRelatedLevel[nGloCount] + 1
  Set nLevelStart = nLevelStart + 1
  Set nGloCount = nGloCount + 1

Function: GetDirectlyRelatedObjects
Description: Lists data concerning either ancestor or dependent objects for a given object into four separate receive arrays depending upon the bDependent boolean (If FALSE, ancestors... If TRUE, dependents) only if the related objects are directly associated (or one level removed) from the given object.

Returns
Number:
Parameters
Number: nObject
Number: nStart
Boolean: bDependent
String: strBaseRelation
Receive Number: nRelated[*]
Receive Number: nRelatedType[*]
Receive String: strObjRelation[*]

Local variables
String: sGloObjectID
String: strObjLoc

Actions
Set sGloCSV = sGloObjAssociations[nObject]
While sGloLength(sGloCSV) > 1
  Set sObjectID = ReadCSV()  
  Call sStrLeft(sObjectID, sStrScan(sObjectID, '.'), sStrLoc)  
  Call sStrLop(sStrLoc)  
  Set nRelated[nStart] = sStrToNumber(sStrLoc)  
  Set nRelatedType[nStart] = nGloObjType[nRelated[nStart]]
Call sStrRight(sObjectID, 1, strObjRelation[nStart])
If (bDependent and strObjRelation[nStart] > 'Z')
  or (not bDependent and strObjRelation[nStart] < 'Z')
If not bDependent
  If strBaseRelation = 'A' or ((strBaseRelation = 'O' or strBaseRelation = 'M' or strBaseRelation = 'X') and strObjRelation[nStart] = 'A')
    Set strObjRelation[nStart] = strBaseRelation
Set nStart = nStart + 1
Set nRelated[nStart] = -1
Return nStart

Function: GetNextAttributePtr
Description: Returns a starting point for a new link list of object attributes.

Returns
Number:
Parameters
Local variables
Number: nAttrCount

Actions
Set nAttrCount = 1
While nGloAttrNextPtr[nAttrCount] != 0
  Set nAttrCount = nAttrCount + 1
Set nGloAttrNextPtr[nAttrCount] = -1
Return nAttrCount
Function: GelTypeSpecifics
Description: Determines whether an attribute type (e.g., CHAR, DATE) requires length and scale and calls a dialog box for the user to enter one or both fields if required by SQL.
Returns
Parameters
Local variables
Actions
  If colType != 'Decimal' and
  colType != 'Character Field' and
  colType != 'Variable Length Field'
  Call SalIClearField(colLength)
  Call SalIClearField(colScale)
  Else
    If not bSpecificsDialog
      If SalIsNull(colLength)
        Set colLength = -1
      End If
      If colLength < 1
        Set colLength = -1
      End If
    Else
      Set nLength = colLength
      Set nScale = colScale
    End If
    If SalModalDialog(digTypeSpecifics,digDetails)
      Set colScale = nScale
      If nScale < -1
        Set nScale = 0
      End If
      Set colLength = nLength + nScale
    Else
      If nLength < -1
        Call SalIClearField(colType)
        Return FALSE
      End If
    End If
  Call AppendSettingsToTypeO
Function: HideEntity
Description: Hides an object displayed in the workspace.
Returns
Parameters
Window Handle: hWndDomain
Local variables
Actions
  Call SalHideWindow(hWndItem)
  Call SalHideWindow(hWndDomain)
Function: IsAlphaNumeric
Description: Determines whether or not a given string consists entirely of alphanumeric characters. Returns TRUE if the string is alphanumeric, FALSE otherwise. Will also replace spaces in the string with an underscore.
Returns
Boolean:
Parameters
Receive string: strName
Local variables
String: strNameCheck
Number: nChar
Actions
  Call ReplaceChar(strName,SPACE,'_')
  Set strNameCheck = strName
  While strNameCheck I= null
    Set nChar = SalStrLen(strNameCheck)
    If (nChar > 47 and nChar < 58) or
       (nChar > 64 and nChar < 91) or
       (nChar > 96 and nChar < 123) or
       nChar = 95
      Else
        Call SalMessageBox("This entry must be alphanumeric.",APPNAME,MB_OK|MB_iconAsterisk)
        Return FALSE
      End If
    Return IsNotReserved(strName)
Function: IsAncestor
Description: Determines whether one object is the ancestor of the other.
Returns
Boolean:
Parameters
Number: nAssoclate
Number: nObject
Local variables
Boolean: bAncestor

Actions
Set bAncestor = FALSE
Call GetGlobRelatedObjects(nObject,FALSE,
    nGloRelated,nGloRelatedLevel,nGloRelatedType,strGloObjRelation)
Set nGloCount = 1
While nGloRelated[nGloCount] != -1
    If nGloRelated[nGloCount] = nAssoclate
        Set bAncestor = TRUE
    Set nGloCount = nGloCount + 1
Return bAncestor

Function: IsDependent
Description: Determines whether one object is dependent upon the other.
Returns
Boolean:
Parameters
Number: nAssoclate
Number: nObject
Local variables
Boolean: bDependent

Actions
Set bDependent = FALSE
Call GetGlobRelatedObjects(nObject,TRUE,
    nGloRelated,nGloRelatedLevel,nGloRelatedType,strGloObjRelation)
Set nGloCount = 1
While nGloRelated[nGloCount] != -1
    If nGloRelated[nGloCount] = nAssoclate
        Set bDependent = TRUE
    Set nGloCount = nGloCount + 1
Return bDependent

Function: IsNotReserved
Description: Determines whether a given string is on the list of SQL reserved words (read in at the start of the program into strGloSQLReservedWords[*] from the file "RESERVE.SQL"). Return TRUE if the string is not on the list, FALSE otherwise.
Returns
Boolean:
Parameters
String: strWord
Local variables
Number: nReserveCount

Actions
Set nReserveCount = 0
Call StrToStr(strWord,strWord)
While strGloSQLReservedWords[nReserveCount] != NULL
    If strGloSQLReservedWords[nReserveCount] = strWord
        Call ErrorMessageBox("""strWord""") is a SQL reserved word. Please change your entry.' , APPNAME, MB_OK, MB_iconAsterisk)
        Return FALSE
    Else
        Set nReserveCount = nReserveCount + 1
End If
Return TRUE

Function: KillFocusColor
Description: Changes the color of an object which has just lost focus so that it no longer appears to have focus.
Returns
Parameters
Number: nColorBkgd
Number: nColorTexl
Local variables
String: strEntityName

Actions
Call SaIColorSet(hWndItem, COLOR_indexWindow, nColorBkgd)
Call SaIColorSet(hWndItem, COLOR_indexWindowText, nColorTexl)

Function: LoadHandles
Description: Loops through a window looking for a particular object type and records a handle
for each object of that type to a given array.

Returns

Parameters

Window Handle: hWndCurrent
Window Handle: hWndArray[50]
Number: nType

Local variables

Window Handle: hWndChild

Actions

Set hWndChild = SaIGetFirstChild(hWndCurrent,nType)
Set nGloCount = 0
Loop
  If hWndChild = hWndNULL
    Break
  Else
    Call SaIHideWindow(hWndChild)
    Set hWndArray(nGloCount) = hWndChild
    Set hWndChild = SaIGetNextChild(hWndChild,nType)
    Set nGloCount = nGloCount + 1
  End If

Function: NoDetailErrors

Description: Loops through the Object Attributes table window on digDetails and validates that each entered attribute has no errors. Returns TRUE if no errors are found, FALSE otherwise.

Returns

Boolean:

Parameters

local variables

Number: nNameCount
String: strCurrentName
Number: nKeyResponse

Actions

Set nAttributeTbIRow = TBL_MinRow
While SaITblFindNextRow(hWndForrn,nAttributeTbIRow,0,0)
  Call SaITblSetContext(hWndForrn,nAttributeTbIRow)
  If SallsNull(colFrom)
    If SallsNull(colName)
      Set colName = 'unnamed'
      Call SaITblSetFocusCell(hWndForrn,nAttributeTbIRow,colName,-1,-1)
      Call SaITblKillEdit(hWndForrn)
      Call SaIMessageBox('Please provide a name for this attribute.',
        APPNAME,MB_OKMB_iconAsterisk)
      Call SaITblSetFocusCell(hWndForrn,nAttributeTbIRow,colName,-1,-1)
      Return FALSE
    Else
      If nAttributeTbIRow > 0
        Set strCurrentName = colName
        Set nNameCount = 0
        While nNameCount < nAttributeTbIRow
          Call SaITblSetContext(hWndForrn,nNameCount)
          If colName = strCurrentName
            Call SaIMessageBox('Duplicate Attribute Found. Please Make Sure That Each Attribute for
              \"\"[\"\"lrrnObjMgr.strO_IDII\"]\"\" has a unique name.',
                APPNAME,MB_OKMB_iconAsterisk)
            Call SaITblSetFocusCell(hWndForrn,nAttributeTbIRow,colName,-1,-1)
            Return FALSE
          Else
            Set nNameCount = nNameCount + 1
            Call SaITblSetContext(hWndForrn,nAttributeTbIRow)
          End If
        End While
        If SallsNull(colKey)
          Set nKeyResponse = SaIMessageBox('Is the \"\"[\"\"colName\"]\"\" a key attribute? (ie. Is the \"\"[\"\"colName\"]\"\" used to identify
            a given \"\"lrrnObjMgr.strO_IDII\"]\"\"'),
            APPNAME,MB_YESNoMB_iconQuestionMB_defButton2)
          If nKeyResponse = IDYES
            Set colKey = 'Yes'
            Set colRequired = 'Yes'
            Set colIndexed = 'Yes'
          ElseIf nKeyResponse = IDNO
            Set colKey = 'No'
          If nKeyResponse = IDCANCEL
        End If
      End If
    End If
  End If
End While
Function: NoDuplicateDependentAttributes
Description: Co-ordinates the process of verifying that no attributes belonging to a given object are duplicated in any dependents.
Returns: Boolean:
Parameters:
Number: nObject
String: strAssociationType
Local variables:
Number: nDupeCount
Number: nDupeRelated
Number: nDupeRelatedDummy
String: strDupeRelatedDummy
Actions:
Call SalWaitCursor(TRUE)
Set strMessage = "Checking for duplicate attributes...
Set bGloDupeCheck = TRUE
Call GetAllRelatedObjects(nObject,TRUE, nDupeRelated,nDupeRelatedDummy,nDupeRelatedDummy,strDupeRelatedDummy)
Set strGloObjRelation[0] = strAssociationType
Set nDupeCount = 0
While nDupeRelated[nDupeCount] != -1
  Set nGloDetailObject = nDupeRelated[nDupeCount]
  If not SalModaIDialog(digDetails,hWndForm)
    If bGloDetailOK
      Call SalMessageBox(strAssoclationType "may not be associated with "{strGloObjRelation} because the
        "{strGloDupeAttr} attribute belonging to "{strGloID}nGloDetailObject\} would be duplicated.
      APPNAME,MB_OK(MB_iconAsterisk)
    Set bGloDupeCheck = FALSE
    Call SalWaitCursor(FALSE)
Return FALSE
Set nDupeCount = nDupeCount + 1
Set bGloDupeCheck = FALSE
Set strMessage = ""
Description: Verifies that associations for a given object do not result in duplicate dependents.

Returns: Boolean.

Parameters:
- Number: nObject
- Receive String: strDuplicates

Local variables:
- Boolean: bNoDuplicate
- Number: nDupeCount

Actions:
- Set bNoDuplicate = TRUE
- Call GetAllRelatedObjects(nObject, TRUE, nGloRelated, nGloRelatedLevel, nGloRelatedType, strGloObjRelation)
- Set nGloCount = 1
- Set strDuplicates = NULL
- While nGloRelated[nGloCount] != -1
  - Set nDupeCount = nGloCount + 1
  - While nGloRelated[nDupeCount] != -1
    - If nGloRelated[nGloCount] = nGloRelated[nDupeCount]
      - Set bNoDuplicate = FALSE
      - Set strDuplicates = strDuplicates + [strGloObjID[nGloRelated[nGloCount]]]
    - Set nDupeCount = nDupeCount + 1
  - Set nGloCount = nGloCount + 1
- If not bNoDuplicate
  - Call SalStrLog(strDuplicates)
  - Call SalStrLog(strDuplicates)
- Return bNoDuplicate

Function: NoDuplicateErrors

Description: Verifies that no attributes for a given object (currently listed in the digDetails window) are duplicates of one another.

Returns: Boolean.

Parameters:
- Local variables:
  - Number: nNameCount
  - String: strCurrentName
  - Number: nKeyResponse

Actions:
- Set nAttributeTblRow = TBL_MinRow
- While SalTbiFindNextRow(hWndForm, nAttributeTblRow, 0, 0)
  - Call SalTbiSetContext(hWndForm, nAttributeTblRow)
  - If nAttributeTblRow > 0
    - Set strCurrentName = colName
    - Set nNameCount = 0
    - While nNameCount < nAttributeTblRow
      - Call SalTbiSetContext(hWndForm, nNameCount)
      - If colName != strCurrentName and colName != 'undefined'
        - Set strGloDupeAttr = colName
        - Return FALSE
      - Else
        - Set nNameCount = nNameCount + 1
        - Call SalTbiSetContext(hWndForm, nAttributeTblRow)
  - Return TRUE

Function: NoErrorsFound

Description: Sends a MSG_Check message to all children of hWndForm (the top level window that currently has focus) for purposes of error checking. If an error is found, the message is displayed in a message box, and FALSE is returned; otherwise TRUE is returned.

Returns: Boolean.

Parameters:
- Local variables:

Actions:
- Set strGloErrorMessage = NULL
- Call SalSendMsgToChildren(hWndForm, MSG_Check, 0, 0)
- If strGloErrorMessage != NULL
  - Call SalMessageBox(strGloErrorMessage, APPNAME, MB_OK|MB_ICONASTERISK)
  - Return FALSE
- Else
  - Return TRUE
Function: OpenOSAMFile
Description: Reads a "sam" file into the object and attribute arrays and returns the user to screen (0,0)
Returns
Parameters
String: strFileName
Local variables
Number: nCount
Number: nPos
Number: nFldCount
Long String: strField[20]
String: strObject
File Handle: hFile
Actions
Call SalWaitCursor(TRUE)
Set strMessage = "Reading OSAM File: "
If SalFileOpen(hFile,strFileName,OF_ReadOnly)
Call SalStrGetStr(hFile,strObject,5000)
While strObject != DEL
Call SalStrTokenize(strObject,DEL,DEL,strField)
Set nCount = SalStrToNumber(strField[0])
Set strObj_ID[nCount] = strField[1]
Set strObjName[nCount] = strField[2]
Set strObjDesc[nCount] = strField[3]
Set nObjCell[nCount] = SalStrToNumber(strField[4])
Set nObjType[nCount] = SalStrToNumber(strField[5])
Set nObjBar[nCount] = SalStrToNumber(strField[6])
Set nGloObjBar[nCount] = SalStrToNumber(strField[7])
Set nGloObjAttrPtr[nCount] = SalStrToNumber(strField[8])
set strGloObjAssociations[nCount] = strField[9]
Set strObjCRel[nCount] = strField[10]
Set strObjInsert[nCount] = strField[12]
Set strObjUpdate[nCount] = strField[13]
Set strObjDelete[nCount] = strField[14]
Set nFldCount = 0
While nFldCount < 15
Set strField[nFldCount] = NULL
Set nFldCount = nFldCount + 1
If strGloObjDesc[nCount] = SPACE
Set strGloObjDesc[nCount] = NULL
If strGloObjAssociations[nCount] = SPACE
Set strGloObjAssociations[nCount] = NULL
If strGloObjCRel[nCount] = SPACE
Set strGloObjCRel[nCount] = NULL
If strGloObjXRel[nCount] = SPACE
Set strGloObjXRel[nCount] = NULL
If strGloObjInsert[nCount] = SPACE
Set strGloObjInsert[nCount] = NULL
If strGloObjUpdate[nCount] = SPACE
Set strGloObjUpdate[nCount] = NULL
If strGloObjDelete[nCount] = SPACE
Set strGloObjDelete[nCount] = NULL
Set nGloNextPos = nCount
Set nGloCell[nGloNextPos] = nGloObjCell[nCount]
Set nGloAbsPos = SalNumberMod(nGloAbsPos,10)
If nPos = 0
Set nPos = 10
Call SalSendMsg(frmObjMgr.hWndDF[nPos],MSG_Copy,0,0)
Call SalFileGetStr(hFile,strObject,5000)
Call SalFileGetStr(hFile,strObject,5000)
While strObject != DEL
Call SalStrTokenize(strObject,DEL,DEL,strField)
Set nCount = SalStrToNumber(strField[0])
Set strGloAttrName[nCount] = strField[1]
Set bGloAttrKey[nCount] = SalStrToNumber(strField[2])
Set bGloAttrRequired[nCount] = SalStrToNumber(strField[3])
Set bGloAttrIndexed[nCount] = SalStrToNumber(strField[4])
Set nGloAttrType[nCount] = SalStrToNumber(strField[5])
Set nGloAttrLength[nCount] = SalStrToNumber(strField[6])
Set nGloAttrScale[nCount] = SalStrToNumber(strField[7])
Set nGloAttrStatus[nCount] = SalStrToNumber(strField[8])
Set nGloAttrNextPtr[nCount] = SalStrToNumber(strField[9])
Set strGloAttrComments[nCount] = strField[10]
Set strGloAttrCRel[nCount] = strField[12]
Set strGloAttrXRel[nCount] = strField[13]
If strGloAttrName[nCount] = SPACE
  Set strGloAttrName[nCount] = NULL
If strGloAttrComments[nCount] = SPACE
  Set strGloAttrComments[nCount] = NULL
If strGloAttrRules[nCount] = SPACE
  Set strGloAttrRules[nCount] = NULL
If strGloAttrCRel[nCount] = SPACE
  Set strGloAttrCRel[nCount] = NULL
If strGloAttrXRel[nCount] = SPACE
  Set strGloAttrXRel[nCount] = NULL
Call SalFileGetStr(hFile,strObject,5000)
Call SalFileClose(hFile)
Set bGloChanged = FALSE
Call SalPostMsg(pbOrigln,SAM_Click,0,0)
Call SalSetWindowText(hWndForm,APPNAMEI' - 'IstrGloFileName)
Set bGloInitialized = FALSE
Set strMessage = NULL
Call SalWaitCursor(FALSE)

Function: ReadCSV
Description: Strips off and returns the first element of global strGloCSV.
Returns
String:
Parameters
Local variables
Number: nOffset
Number: nLength
Number: nQuote
String: sTemp
Actions
Set nLength = SalStrLength(strGloCSV)
Set nOffset = SalStrScan(strGloCSV::
Call SalStrLeft(strGloCSV,nOffset,sTemp)
Call SalStrRight(strGloCSV,nLength-nOffset-1,strGloCSV)
Return sTemp

Function: ReadReservedWords
Description: Reads a list of SQL reserved words from the RESERVE.SQL file.
Returns
Parameters
Local variables
File Handle: hFile
Actions
Call SalWaitCursor(TRUE)
Set nGloCount = 0
If SalFileOpen(hFile:'RESERVE.SQL',CF_Read)
  While SalFileGetStr(hFile,strGloSQLReservedWord[nGloCount],100)
    Call SalStrLower(strGloSQLReservedWord[nGloCount],strGloSQLReservedWord[nGloCount])
    Set nGloCount = nGloCount + 1
  Call SalFileClose(hFile)
Set strGloSQLReservedWord[nGloCount] = NULL
Call SalWaitCursor(FALSE)

Function: Redraw
Description: Coordinates the repainting of the workspace when the user has made a change to it.
Returns
Parameters
Local variables
Actions
Call SalWaitCursor(TRUE)
Set sMessage = 'Redrawing Workspace...
Set nGloEditPos = 0
Sel bEntityFocus = FALSE
Set nGloResetPos = nGloScreen*10
Set nGloCount = 1
While nGloCount < MAXLINES+1
    Call SalHideWindow(fnnObJMgr.hWndLine[nGloCount])
    Set nGloCount = nGloCount + 1
Call SalSendMsgToChildren(hWndForm,MSG_Hide,0,0)
Call SalSendMsgToChildren(hWndForm,MSG_Reset,0,0)
Call SalSendMsgToChildren(hWndForm,MSG_Gray,0,0)
Call SalSendMsgToChildren(hWndForm,MSG_Show,0,0)
Call ShowLines()
Call RefreshSelectMenu()
Set strMessage = NULL
Call SalWaitCursor(FALSE)

Function: ResetEntily
Description: Resets the regular colors—cyan or gray (for domains) on all visible objects.
Returns
Parameters
Local variables
Actions
Set nGloResetPos = nGloResetPos + 1.
If strGlo_ID[nGloCell[nGloResetPos]] != NULL and nGloCell[nGloResetPos] != 0
    Call SalSetWindowText(hWndItem,strGlo_ID[nGloCell[nGloResetPos]])
If nGloObjType[nGloCell[nGloResetPos]] = DOMAIN
    Call SalColorSet(hWndItem,COLOR_indexWindow,COLOR_Gray)
Else
    Call SalColorSet(hWndItem,COLOR_indexWindow,COLOR_Cyan)
Else
    Call SalClearField(hWndItem)

Function: ResetFileNames
Description: resets buffers for file names (for use with the Common File Dialogs)
Returns
Parameters
Receive String: strFile
Receive String: strPath
String: strExtension
Local variables
Actions
Call SalStrSetBufferLength(strFile,256)
Call SalStrSetBufferLength(strPath,256)
Call SalStrLeft(strGloFileName,SalStrScan(strGloFileName,''),strGloFileName)
Set strGloFileName = strGloFileName[\"\"\"\"strExtension
Call SalStrLeft(strGloFilePath,SalStrScan(strGloFilePath,''),strGloFilePath)
If strGloFilePath = NULL
    Set strGloFilePath = ".\"\"\strExtension
Else
    Set strGloFilePath = strGloFilePath[\"\"\"\"strExtension
If strGloFileName = NULL
    Call SalStrLeft(strGloDefPath,SalStrLength(strGloDefPath)-(SalStrLength(strFile)+1),strGloDefPath)
Else
    Set strGloDefPath = \\

Function: RefreshLabels
Description: Updates the labels which are visible on the workspace to reflect each object's association
(if any) with the object with focus.
Returns
Parameters
Local variables
String: strConnectPos
Number: nConnect1
Number: nConnect2
Number: nPos1
Number: nPos2
Number: nMaxPos
Number: nScanPos
Boolean: bNotCurrent
String: strAssocType
Actions
Set nMaxPos = (nGloScreen*10)+10
Set nPos1 = (nGloScreen * 10) + 1
While nPos1 <= nMaxPos
    Set nConnect1 = SalNumberMod(nPos1, 10)
    If nConnect1 = 0
        Set nConnect1 = 10
    End If
    Call SalNumberToStr(nGloCell[nPos1], 0, strConnectPos)
    Set nScanPos = SaIStrScan(strGloObjAssociations[nGloCell[nGloAbsPos]], '#', strConnectPos)
    If nScanPos = -1
        Set strAssocType = 'SPACE'
        While strAssocType < 'A'
            Call SaIStrMid(strGloObjAssociations[nGloCell[nGloAbsPos]], nScanPos, 1, strAssocType)
            Set nScanPos = nScanPos + 1
        End While
        Call SalSetWindowText(hWndDF[nConnect1], strGloO_ID[nGloCell[nPos1]])
    Else
        Call SalSetWindowText(hWndDF[nConnect1], strGloO_ID[nGloCell[nPos1]])
        If nGloAbsPos = nPos1
            If nGloObjType[nGloCell[nPos1]] = 'DOMAIN'
                Call SalIColorSet(fnnObjMgr.hWndDF[nConnect1], COLOR_IndexWindow, COLOR_Cyan)
            Else
                Call SalPicSetFile(hWndDom[nConnect1], CIRCLE_OFF)
            End If
            nPos1 = nPos1 + 1
        End If
    End If
End While

Function: RefreshSelectMenu
Description: Updates the select menu to reflect the objects currently visible on the screen.

Returns
Parameters
Local variables
Number: nMenuCount, nPos, nOffset
String: strMenuItem
Actions
Set nPos = (nGloScreen * 10) + 1
Set nMenuCount = 0
Set nOffset = 0
Set bGloItemsVisible = FALSE
While nMenuCount < 10
    If nMenuCount = 4
        Set nOffset = 1
    ElseIf nMenuCount = 6
        Set nOffset = 2
    ElseIf nMenuCount = 9
        Set strMenuItem = '180 -'
    Else
        Call SalNumberToStr(nMenuCount + 1, 0, strMenuItem)
        Set strMenuItem = strMenuItem & nMenuCount
        If nGloCell[nPos+nMenuCount] > 0
            Set nMenuItem = GetMenuItemID(hWndGloSubMenu, nMenuCount+nOffset, strMenuItem)
            Set bGloItemsVisible = TRUE
            Call SalStrProper(strString, strMenuItem)
            Call ChangeMenu(hWndGloSubMenu, nMenuCount+nOffset, strMenuItem, nMenuCount+nOffset, MF_ByPosition | MF_String | MF_Change)
        End If
    End If
End While
Call SalIDrawMenuBar(fnnObjMgr)

Function: ReplaceChar
Description: scans a string and replaces a given character with a second character.

Returns
Parameters
Receive String: strName
String: strChar
String: strReplacement
Local variables
Actions
While SaIStrScan(strName, strChar) 1 = -1
    Call SalStrReplace(strName, SaIStrScan(strName, strChar), 1, strReplacement, strName)
End While

Function: RestoreDetails
Description: copies object attributes held in temporary variables to a given object's attribute link list.

Returns
Parameters
Actions
Boolean:
Parameters
Number: nObjectID
Local variables
Number: nNextPtr
Number: nLastPtr
Number: nCopyCount
Actions
If nGloAttrPtr[nObjectID] = 0
Return TRUE
Else
Set nCopyCount = 0
Set nNextPtr = nGloObjAttrPtr[nObjectID]
While nNextPtr != -1 and strGloAttrCopyName[nCopyCount] != NULL
Set strGloAttrName[nNextPtr] = strGloAttrCopyName[nCopyCount]
Set bGloAttrKey[nNextPtr] = bGloAttrCopyKey[nCopyCount]
Set bGloAttrRequired[nNextPtr] = bGloAttrCopyRequired[nCopyCount]
Set bGloAttrIndexed[nNextPtr] = bGloAttrCopyIndexed[nCopyCount]
Set nGloAttrType[nNextPtr] = nGloAttrCopyType[nCopyCount]
Set nGloAttrLength[nNextPtr] = nGloAttrCopyLength[nCopyCount]
Set nGloAttrScale[nNextPtr] = nGloAttrCopyScale[nCopyCount]
Set strGloAttrComments[nNextPtr] = strGloAttrCopyComments[nCopyCount]
Set nCopyCount = nCopyCount + 1
End
If nGloAttrNextPtr[nNextPtr] != -1
Set nGloAttrNextPtr[nNextPtr] = GetNextAttributePtr()
Set nLastPtr = nNextPtr
Set nNextPtr = nGloAttrNextPtr[nNextPtr]
If nNextPtr != -1
Set nGloAttrNextPtr[nLastPtr] = -1
Set nGloAttrNextPtr[nNextPtr] = 0
Return TRUE

Function: RetrieveCrossProductAttributes
Description: Generates the SQL "group by" fields when building SQL stored statements.
Returns
String:
Parameters
Number: nObject
Local variables
Number: nNextPtr
Boolean: bAttributeFound
Number: nRelated[*]
Number: nLevel[*]
Number: nType[*]
String: strRelation[*]
Number: nCount
String: strCrossProduct
Actions
Set strCrossProduct = NULL
Call GetAllRelatedObjects(nObject,FALSE,nRelated,nLevel,nType,strRelation)
Set nCount = 1
While nRelated[nCount] != -1
Call SalStrUpper(strRelation[nCount],strRelation[nCount])
If strRelation[nCount] = 'X'
Set strRelation[nCount] = 'A'
If nGloObjAttrPtr[nRelated[nCount]] != 0
Set nNextPtr = nGloObjAttrPtr[nRelated[nCount]]
While nNextPtr != -1
If nGloAttrStatus[nNextPtr] = ACTIVE and bGloAttrKey[nNextPtr]
Set strCrossProduct = strCrossProduct||strGloAttrName[nNextPtr]||', '
Set nNextPtr = nGloAttrNextPtr[nNextPtr]
Set nCount = nCount + 1
End
If strCrossProduct != NULL
Call SalStrLeft(strCrossProduct,SalStrLength(strCrossProduct)-2,strCrossProduct)
Return strCrossProduct

Function: RetrieveObjectAttributes
Description: Populates the digDetail table with the attributes belonging to a given object.
Returns
- 93 -
Boolean:
Parameters
Number: nObject
Number: nRelLevel
String: strAssocType
Local variables
Number: nNextPtr
Number: nNewRow
String: strOrigAssocType
Boolean: bAttributeFound
Actions
Call SaIStrUpper(strAssocType,strAssocType)
Set strOrigAssocType = strAssocType
If strAssocType = 'X'
    Set strAssocType = 'A'
If nGloObjAttrPtr[nObject] = 0 or
    (nRelLevel > 1 and strAssocType = 'G' and strOrigAssocType = 'X')
    Set bAttributeFound = FALSE
Else
    Set nNextPtr = nGloObjAttrPtr[nObject]
    Set bAttributeFound = FALSE
While nNextPtr != -1
    If nGloAttrStatus[nNextPtr] = ACTIVE and
        (((bGloAttrKey[nNextPtr] and nRelLevel = 1 or strOrigAssocType = 'X'))
         or nObject = nGloDetailObject or strAssocType = 'G')
        Set bAttributeFound = TRUE
    Set nNewRow = SaITblInsertRow(hWndForm, TBL_MaxRow )
    Call SaITblSetContext(hWndForm,nNewRow )
    Call SaITblSetRowFlags(hWndForm,nNewRow,ROW_New,FALSE)
    Set colName = strGloAttrName[nNextPtr]
    If nObject != nGloDetailObject
        Set colFrom = strGloO_ID[nObject]
        Set colTableName = strGloObjName[nObject]
        Set colAssocType = strGloAttrType[nNextPtr]
        Call SaIColorSet(colName,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colFrom,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colTableName,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colAssocType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
        Call SaIColorSet(colIndexType,COLOR_indexCellText,COLOR_DarkGreen)
    Else
        If not bGloDetailOK and colName = strGloDupeAttr
            Call SaITblSetFocusCell(hWndForm,nNewRow,colName,-1,-1)
        If bGloAttrKey[nNextPtr] and nObject = nGloDetailObject or strAssocType != 'A'
            Set colKey = 'Yes'
        Else
            Set colKey = 'No'
        If bGloAttrRequired[nNextPtr] and nObject = nGloDetailObject or strAssocType != 'A'
            Set colRequired = 'Yes'
        Else
            Set colRequired = 'No'
        If bGloAttrIndexed[nNextPtr] and nObject = nGloDetailObject or strAssocType != 'A'
            Set colIndexed = 'Yes'
        Else
            Set colIndexed = 'No'
        Set colItemType = nGloAttrType[nNextPtr]
        Set colComments = strGloAttrComments[nNextPtr]
        Set colCRel = nGloAttrCRel[nNextPtr]
        Set colXRel = nGloAttrXRel[nNextPtr]
        Call SaIStrReplace(strColType[colItemType],SaIStrScan(strColType,colItemType),'&'),1,NULL,colType)
        Call SaIStrMid(TYPE_CHOICES,colItemType,1,strCheck)
        Set colLength = nGloAttrLength[nNextPtr]
        Set colScale = nGloAttrScale[nNextPtr]
        Set nScale = colScale
        If nScale = -1
            Set nScale = 0
Set nLength = colLength-nScale 
Call AppendSettingsToTypeO 
Set nNextPtr = nGloAttrNextPtr[nNextPtr] 
If not bAttributeFound and (nRelLevel = 1 or strAssocType = 'G') and nObject != nGloDetailObject 
Set nNewRow = SetTlblinserRow(hWndForm, TBL_MaxRow) 
Call SetTlbSelContext(hWndForm, nNewRow) 
Call SetTlbSelRowFlags(hWndForm, nNewRow, ROW_New, FALSE) 
Call SetColorSel(colObject, COLOR_IndexCellText, COLOR_Red) 
Call SetColorSel(colFrom, COLOR_IndexCellText, COLOR_Red) 
Set colName = '(undefined)' 
Set colFrom = strGloO_ID[nObject] 
Set colAssocType = strAssocType 
Sel col Key = NULL 
Set colRequired = NULL 
Set colIndexed = NULL 
Set colType = NULL 
Return bAttributeFound 

Function: SaveDetalis 
Description: writes the base attributes (attributes not inherited from other objects) for a given object to that object's attribute link list. 

Returns: Boolean 
Parameters: 
Local variables: 
Number: nNextPtr 
Number: nLastPtr 

Actions: 
If nGloObjAttrPtr[nGloCell[nGloAbsPos]] = 0 
  Set nGloObjAttrPtr[nGloCell[nGloAbsPos]] = GetNextAttributePtr() 
Set nNextPtr = nGloObjAttrPtr[nGloCell[nGloAbsPos]] 
Set nAttributeTblRow = TBL_MinRow 
While SetTlbFindNextRow(hWndForm, nAttributeTblRow, 0,0) 
  Call SetTlbSelContext(hWndForm, nAttributeTblRow) 
  If colFrom = NULL 
    Set bGloAttrName[nNextPtr] = colName 
    If colKey = 'Yes' 
      Set bGloAttrKey[nNextPtr] = TRUE 
    Else 
      Set bGloAttrKey[nNextPtr] = FALSE 
    If colRequired = 'Yes' 
      Set bGloAttrRequired[nNextPtr] = TRUE 
    Else 
      Set bGloAttrRequired[nNextPtr] = FALSE 
    If collindexed = 'Yes' 
      Set bGloAttrIndexed[nNextPtr] = TRUE 
    Else 
      Set bGloAttrIndexed[nNextPtr] = FALSE 
    Set bGloAttrStatus[nNextPtr] = ACTIVE 
    Set bGloAttrType[nNextPtr] = colItemType 
    Set bGloAttrLength[nNextPtr] = colLength 
    Set bGloAttrScale[nNextPtr] = colScale 
    Set strGloAttrComments[nNextPtr] = colComments 
    Set strGloAttrCRel[nNextPtr] = colCRel 
    Set strGloAttrXRel[nNextPtr] = colXRel 
    If nGloAttrNextPtr[nNextPtr] = -1 
      Set nGloAttrNextPtr[nNextPtr] = GetNextAttributePtr() 
    Set nLastPtr = nNextPtr 
    Set nNextPtr = nGloAttrNextPtr[nNextPtr] 
    If nNextPtr = -1 
      Set nGloAttrNextPtr[nLastPtr] = -1 
    Set nGloAttrNextPtr[nNextPtr] = 0 
    Return TRUE 

Function: SaveOSAMFile 
Description: writes object and attribute arrays to a "sam" file. 

Returns: 
Parameters: 
String: strFileName
Local variables
String: strCount
String: strObjCell
String: strObjType
String: strObjVBar
String: strObjAttrPlr
File Handle: hFile
String: strAttrKey
String: strAttrRequired
String: strAttrIndexed
String: strAttrLength
String: strAttrScale
String: strAttrStatus
String: strAttrNextPtr
Number: nNextPtr
String: strNext

Actions
Call SaISendMsgToChildren(hWndForm, SAM_FieldEdit, 0, 0)
If SaIFileOpen(hFile, strFileName, OF_Create, OF_Write)
    Call SaWaitCursor(TRUE)
    Set strMessage = "Saving Workspace to OSAM" File: '[[strFileName]]'...
Set nGloCount = 1
While nGloObjType[nGloCount] != DELETE
    Call SaINumberToStr(nGloObjType[nGloCount], 0, strObjType)
    Call SaINumberToStr(nGloObjCell[nGloCount], 0, strObjCell)
    Call SaINumberToStr(nGloObjHBar[nGloCount], 0, strObjHBar)
    Call SaINumberToStr(nGloObjVBar[nGloCount], 0, strObjVBar)
    Call SaINumberToStr(nGloObjAttrPlr[nGloCount], 0, strObjAttrPlr)
    If strGloObjDesc[nGloCount] = NULL
        Set strGloObjDesc[nGloCount] = SPACE
    If strGloObjAssociations[nGloCount] = NULL
        Set strGloObjAssociations[nGloCount] = SPACE
    If strGloObjCRel[nGloCount] = NULL
        Set strGloObjCRel[nGloCount] = SPACE
    If strGloObjXRel[nGloCount] = NULL
        Set strGloObjXRel[nGloCount] = SPACE
    If strGloObjInsert[nGloCount] = NULL
        Set strGloObjInsert[nGloCount] = SPACE
    If strGloObjUpdate[nGloCount] = NULL
        Set strGloObjUpdate[nGloCount] = SPACE
    If strGloObjDelete[nGloCount] = NULL
        Set strGloObjDelete[nGloCount] = SPACE
        Call SaIFilePutStr(hFile, strCount)
        Call SaIFilePutStr(hFile, strObjCell)
        Call SaIFilePutStr(hFile, strObjType)
        Call SaIFilePutStr(hFile, strObjHBar)
        Call SaIFilePutStr(hFile, strObjVBar)
        Call SaIFilePutStr(hFile, strObjAttrPlr)
        Call SaIFilePutStr(hFile, strGloObjDesc)
        Call SaIFilePutStr(hFile, strGloObjAssociations)
        Call SaIFilePutStr(hFile, strGloObjCRel)
        Call SaIFilePutStr(hFile, strGloObjXRel)
        Call SaIFilePutStr(hFile, strGloObjInsert)
        Call SaIFilePutStr(hFile, strGloObjUpdate)
        Call SaIFilePutStr(hFile, strGloObjDelete)
Set nGloCount = nGloCount + 1
Call SaIFilePutStr(hFile, DEL)
Set nGloCount = 1
While nGloObjType[nGloCount] != DELETE
    If nGloObjAttrPlr[nGloCount] != 0
        Set nNextPtr = nGloObjAttrPlr[nGloCount]
    While nNextPtr != -1
        Call SaINumberToStr(nNextPtr, 0, strNext)
        Call SaINumberToStr(nNextPtr, 0, strAttrKey)
        Call SaINumberToStr(nNextPtr, 0, strAttrRequired)
        Call SaINumberToStr(nNextPtr, 0, strAttrIndexed)
        Call SaINumberToStr(nNextPtr, 0, strAttrType)
        Call SaINumberToStr(nNextPtr, 0, strAttrLength)
        Call SaINumberToStr(nNextPtr, 0, strAttrScale)
        Call SaINumberToStr(nNextPtr, 0, strAttrStatus)
        Call SaINumberToStr(nNextPtr, 0, strAttrNextPtr)
If strGloAttrName[nNextPtr] = NULL
    Set strGloAttrName[nNextPtr] = SPACE
If strGloAttrComments[nNextPtr] = NULL
    Set strGloAttrComments[nNextPtr] = SPACE
If strGloAttrRules[nNextPtr] = NULL
    Set strGloAttrRules[nNextPtr] = SPACE
If strGloAttrCrRel[nNextPtr] = NULL
    Set strGloAttrCrRel[nNextPtr] = SPACE
If strGloAttrXRel[nNextPtr] = NULL
    Set strGloAttrXRel[nNextPtr] = SPACE
If strAttrLength = NULL
    Set strAttrLength = '0'
If strAttrScale = NULL
    Set strAttrScale = '0'
Call SaFllePutStr(hFile,
DELIstrGloAttrRules[nNextPtr]I[DELIstrGloAttrCrRel[nNextPtr]I[DELIstrGloAttrXRel[nNextPtr]])
Set nNextPtr = nGloAttrNextPtr[nNextPtr]
Set nGloCount = nGloCount + 1
Call SaFllePutStr(hFile,DEL)
Call SaISetWlndowText(hWndForm,APPNAMEII'strGloFlleName)
Call SaIFlleClose(hFlle)
Set bGloChanged = FALSE
Set strMessage = NULL
Call SaIWaitCursor(FALSE)

Function: SetDomainGray
Description: Repaints a domain when that domain loses focus.
Returns
Parameters
Window Handle: hWndDomain
Local variables
Actions
If SaIColorGet(hWndltem,COLORJndexWindow) I= COLOR_Gray
    Call K11FocusCoIor(COLOR_Cyan,COLOR_Black)
Else
    Call SaIPicSetFlie(hWndDomain,CIRCLE_OFF)

Function: SetEntityFocus
Description: Repaints an object visible on the screen when that object gets focus.
Returns
Parameters
Number: nEntity
Window Handle: hWndDomain
Local variables
Actions
Set nGloScreenPos = nEntity
Set bEntityFocus = TRUE
Set nGloAbsPos = (nGloScreen*10)+nGloScreenPos
Call RefreshLabelsO
If SaIColorGet(hWndltem,COLOR_IndexWindow) I= COLOR_Gray
    Call SetFocusColor(COLOR_ Yellow,COLOR_Black)
Else
    Call SetFocusColor(COLOR_Gray,COLOR_Black)
    Call SaIPicSetFlie(hWndDornaln,CIRCLE_ON)

Function: SetFocusColor
Description: Repaints a data field a certain foreground and background and sends a gray message to all children of the form.
Returns
Parameters
Number: nColorBkgd
Number: nColorText
Local variables
String: strEntltyName
Actions
Call SaISendMsgToChildren(hWndForm,MSG _Gray,O,O)
Call SaIColorSel(hWndltem,COLOR_lndexWindOW,nColorBkgd)
Call SalColorSel(hWndllem, COLOR_IndexWindowText, nColorText)

Function: SetPositionBlock
Description: Blacks out an object position for placing an object on the workspace if that position is already taken by another object. (called from dlgAddDialog)

Returns
Parameters
Local variables
Actions
If SalColorGet(hWndItem, COLOR_IndexWindow) = COLOR_Yellow
   Call KillFocusColor(COLOR_Cyan, COLOR_Black)

Function: SetPositionFocus
Description: Determines which object position has been selected for placing an object on the workspace (called from dlgAddDialog)

Returns
Parameters
Local variables
Actions
   Call SetFocusColor(COLOR_Yellow, COLOR_Black)
   Call SalGetWindowText(hWndItem, strPos, 2)
   Return SalStrToNumber(strPos)

Function: SetPositionValue
Description: Populates a given position field (hWndItem) with a given number (1..10). Returns TRUE if the position is available, FALSE otherwise.

Returns
Parameters
Number: nPos
Boolean: bPositionFilled
Local variables
String: strPos
Actions
If SalIsWindowVisible(frmObjMgr.hWndDF(nPos)) = bPositionFilled
   Call SalColorSet(hWndItem, COLOR_IndexWindow, COLOR_Black)
   Call SalDisableWindow(hWndItem)
   Return FALSE
Else
   Call SalNumberToStr(nPos, 0, strPos)
   Call SalSetWindowText(hWndItem, strPos)
   Return TRUE

Function: ShowDomain
Description: Shows the corresponding object circle (representing a domain) for given screen position. If that object is a domain.

Returns
Parameters
Window Handle: hWndDomain
Local variables
String: strText
Actions
If SalIsNotNull(hWndItem)
   Call SalShowWindow(hWndItem)
   If SalColorGet(hWndItem, COLOR_IndexWindow) = COLOR_Gray
      Call SalShowWindow(hWndDomain)
   Call SalGetWindowText(hWndItem, strText, 8)
   If strText = frmObjMgr.strO_ID
      Call SalPostMsg(hWndItem, SAM_SetFocus, 0, 0)

Function: ShowLines
Description: loops through the objects currently visible in the workspace and coordinates the redrawing of lines and association (directional) labels.

Returns
Parameters
Local variables
String: strConnectPos
Number: nConnect1
Number: nConnect2
Number: nPos1
Number: nPos2
Number: nMaxPos
Number: nScanPos
Boolean: bNotCurrent
String: strAssocType

Actions
Set nMaxPos = (nGloScreen*10)+10
Set bNotCurrent = FALSE
If SalNumberTruncate(nGloAbsPos/10,0,0) = nGloScreen
and nGloAbsPos != nMaxPos
Set bNotCurrent = TRUE
Set nPos1 = (nGloScreen*10)+1
While nPos1 <= nMaxPos
Call SalNumberToStr(nGloCell[nPos1],0,strConnectPos)
Set nConnect1 = SalNumberMod(nPos1,10)
If nConnect1 = 0
Set nConnect1 = 10
Set nPos2 = nPos1+1
While nPos2 <= nMaxPos
Set nScanPos =
SallStrScan(strGloObjAssociations[nGloCell[nPos2]],#||strConnectPos)
If nGloCell[nPos2] > 0 and nScanPos != -1
Set nConnect2 = SalNumberMod(nPos2,10)
If nConnect2 = 0
Set nConnect2 = 10
Call SalSendMsg(frmObjMgr.hWndDF[nConnect1],MSG_ShowLine,nConnect2,0)
If nPos1 = nGloAbsPos
Set strAssocType = SPACE
While strAssocType < 'A'
Call SalStrMid(strGloObjAssociations[nGloCell[nPos2]],nScanPos,1,strAssocType)
Set nScanPos = nScanPos+1
If strAssocType > 'Z'
Call SalStrUpper(strAssocType,strAssocType)
Else
Call SalStrLower(strAssocType,strAssocType)
Call SalSetWindowText(hWndDF[nConnect1],strGloO_ID[nGloCell[nPos2]][#||strAssocType][])
Set nPos2 = nPos2+1
Set nScanPos =
SallStrScan(strGloObjAssociations[nGloCell[nGloAbsPos]],#||strConnectPos)
If bNotCurrent and nScanPos != -1
Set strAssocType = SPACE
While strAssocType < 'A'
Call SalStrMid(strGloObjAssociations[nGloCell[nGloAbsPos]],nScanPos,1,strAssocType)
Set nScanPos = nScanPos+1
Call SalSetWindowText(hWndDF[nConnect1],strGloO_ID[nGloCell[nPos1]][#||strAssocType][])
If nGloObjType[nGloCell[nPos1]] = DOMAIN
Call SalSetColor(frmObjMgr.hWndDF[nConnect1],COLOR_IndexWindow,COLOR_Green)
Else
Call SalPicSetFile(hWndOomI10-nConnect1),CIRCLE_CONNECTED)
Set nPos1 = nPos1+1

Function: WordWrap
Description: Wraps comments or error messages being written to an SQL file.
Returns
Parameters
File Handle: hFile
String: strText
String: strDelimiter
String: strHeader
Boolean: bFlip
Local variables
String: strLine
String: strChar
String: strWord(*)
Number: nWord

Actions
If bFlip
Set strText = FlipCase(strText)
Call SalStrTrim(strText,strText)
Call SalstrTokenize(strText,SPACE,SPACE,strWord)
Set nWord = 0
Set strLine = strHeader
Call SaIStrRepeat(SPACE, SaIStrLength(strHeader), strHeader)

While strWord[nWord] = NULL
    While (SaIStrLength(strLine) + SaIStrLength(strWord[nWord]) < 85) and strWord[nWord] = NULL
        Set nWord = nWord + 1
    If strDelimiter = NULL
        Call SaIStrLocate(strLine)
        Call SaIFilePutStr(hFile, strDelimiter || strLine)
    Set strLine = strHeader
End While

Function: WriteSQLToFile
Description: writes SQL (generated by other functions) from temporary variables to the SQL file.

Returns
Parameters
    File Handle: hFile
    Receive String: strText[*]
    Boolean: bProper
    Boolean: bUpper
    Boolean: bLower
    Boolean: bObjects
    Boolean: bWordWrap

Local variables
    Number: nBegin
    Number: nWordCount
    String: strWord[*]
    Number: nTokens
    Boolean: bUndefined
    Boolean: bComments
    String: strComment(5)

Actions
    Set bUndefined = FALSE
    Set bComments = FALSE
    While strText[nBegin] = NULL
        If strText[nBegin] = 'COMMENT'
            Set bComments = TRUE
            Set bWordWrap = TRUE
        Else
            If SaIStrScan(strText[nBegin], '(', undefined) = -1
                Set bUndefined = TRUE
            Else
                If SaIStrScan(strText[nBegin], ')') = -1
                    If bComments
                        Call SaIStrTokenize(strText[nBegin], ',', ',', strComment)
                        Set strText[nBegin] = strComment[0]
                    If bProper
                        Call SaIStrProper(strText[nBegin], strText[nBegin])
                    If bObjects and strText[nBegin] = SPACE
                        Set strText[nBegin] = FlipCase(strText[nBegin])
                    If bUpper
                        Call SaIStrUpper(strText[nBegin], strText[nBegin])
                    If bLower
                        Call SaIStrLower(strText[nBegin], strText[nBegin])
                    If bComments
                        Call SaIFilePutStr(hFile, ''
                    If bWordWrap and strText[nBegin] = SPACE
                        Call WordWrap(hFile, strText[nBegin], NULL, NULL, FALSE)
                    Else
                        Call SaIFilePutStr(hFile, strText[nBegin], 'KEY') = -1
                        Call SaIFilePutStr(hFile, strText[nBegin])
                    Else
                        Call WordWrap(hFile, strText[nBegin], NULL, NULL, FALSE)
                    End If
                    Set strText[nBegin] = NULL
                    Set nBegin = nBegin + 1
            End If
        End If
    End While
    If nBegin > 0
        Call SaIFilePutStr(hFile, NULL)
    End If
    If bUndefined
        Call WordWrap(hFile, NULL)

Undefined Attributes Were Inherited From "[[strGloObjName][nGloDetailObject]]" is incomplete.

dfDelimiter: ERROR: ',FALSE)
Sel
nGIoErrorFound = nGIoErrorFound - 1
Call SaIFIIePutStr(hFlle,dfDelimiter)
Call SaIFllePutStr(hFlle,NULL)

Application Actions
On SAM_AppStartup
Set hWndGloWalt = SaICrealeWlndow(dlgWail,hWndForm)
Call ReadReservedWords()

Form Window: frmObjMgr
Title: OSAM Designer - (untitled)
Icon File: osam.ico

Display Settings
Visible at Design time? No
Automatically Created at Runtime? Yes
Initial State: Normal
Maximizable? No
Minimizable? Yes
System Menu? Yes
Resizable? No
Window Location and Size
Left: 0.0"
Top: 0.0"
Width: 7.0"
Height: 4.948"
Form Size
Width: Default
Height: Default
Number of Pages: Dynamic
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Gray
Menu
Popup Menu: &File
Enabled when: bGloDetailOK
Menu Item: &New
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call FlleNew(TRUE)
Menu Item: &Open
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call OpenOSAMFlle(strGloFllePath)

Else
Call SaIStrScan(strGloFlleName, OFN_FILEMUSTEXIST) = -1
If FileNew(FALSE)
Call OpenOSAMFile(strGloFilePath)
Else
Call SaIStoreApp("notepad.exe",strGloFilePath)

Menu Item: &Save...
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call ResetFileNames(strGloFileName,strGloFilePath,'sam')
If strGloFilePath = NULL -101
If DlgSaveAs(hWndForm,strGloDefPath,APPNAMEII' - Save File','sam','OSAM
Files-' .sam- ONFN_PATHMUSTEXIST|ONFN_HIDEREADONLY|ONFN_OVERWRITEPROMPT,strGloFilePath,strGloFileName)
Else
Call SaveOSAMFile(strGloFilePath)
Set strGloFilePath = strGloFilePath&'.sam'
Call SaveOSAMFile(strGloFilePath)
Menu Item: Save As...
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call ResetFileNames(strGloFileName,strGloFilePath,'sam')
If DlgSaveAs(hWndForm,strGloDefPath,APPNAMEII' - Save File','sam','OSAM
Files-' .sam- ONFN_PATHMUSTEXIST|ONFN_HIDEREADONLY|ONFN_OVERWRITEPROMPT,strGloFilePath,strGloFileName)
Else
Call SaveOSAMFile(strGloFilePath)
Menu Item: Build SQL...
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SQLModelDialog(dlgBuildSQL,hWndForm)
Menu Separator
Menu Item: Exit
Keyboard Accelerator: (none)
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
Set nGloObjFunction = CUT
Set bEntityFocus = FALSE
Call CopyObject(nGloCell[nGloAbsPos],0)
Set nGloEditPos = nGloAbsPos
Call DeleteObject(nGloCell[nGloAbsPos])
Set bGloChanged = TRUE
Menu Item: Copy
Keyboard Accelerator: Ctrl+Ins
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
Set nGloObjFunction = COPY
Set nGloEditPos = 0
Call CopyObject(nGloCell[nGloAbsPos],0)
Menu Item: Paste
Keyboard Accelerator: Shift+Ins
Menu Settings
Enabled when: nGloObjFunction = DELETE
Checked when:
Menu Actions
Call SQLDialog(dlgAddObject,hWndForm)
Menu Item: Clear
Keyboard Accelerator: Del
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
If SalMessageBox("You are about to delete the [strGloO_ID[nGloCell[nGloAbsPos]]] object and all associations involving the [strGloO_ID[nGloCell[nGloAbsPos]]] object. Once deleted, an object may not be recovered. Are you sure you want to continue?", APPNAME, MB_YesNo[MB_IconExclamation][MB_DefButtonText] = IDYES
Set bEntityFocus = FALSE
Set nGloObjFunction = DELETE
Set nGloEditPos = nGloAbsPos
Call DeleteObject(nGloCell[nGloAbsPos])
Set bGloChanged = TRUE
Popup Menu: &Object
Enabled when: nGloEntityFocus
Keyboard Accelerator: (none)
Menu Settings
Enabled when: nGloObjFunction = CUT
Checked when:
Menu Actions
Set bGloAddObject = TRUE
Call SalModalDialog(dlgAddObject, hWndForm)
Menu Item: &Find...
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalModalDialog(dlgFindObject, hWndForm)
Menu Separator
Menu Item: &Associate...
Keyboard Accelerator: (none)
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
Call SalModalDialog(dlgAssociateRegular, hWndForm)
Menu Item: &Disassociate...
Keyboard Accelerator: (none)
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
Call SalModalDialog(dlgDisassociate, hWndForm)
Menu Separator
Menu Item: Define &Attributes...
Keyboard Accelerator: (none)
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
If nGloObjType[nGloCell[nGloAbsPos]] = ENTITY_Composite or nGloObjType[nGloCell[nGloAbsPos]] = ENTITY_CrossProduct
Call SalMessageBox("The [frmObjMgr.strO_ID] entity is defined via the regular entities associated with it.", APPNAME, MB_OK[MB_IconAsterisk])
Else
Set nGloDetailsObject = nGloCell[nGloAbsPos]
Set bGloDetailsOK = TRUE
Loop
If SalModalDialog(dlgDetails, hWndForm)
Set bGloDetailsOK = SalModalDialog(dlgDetailsCheck, hWndForm)
If bGloDetailsOK
Break
Menu Item: Define &Rules...
Keyboard Accelerator: (none)
Menu Settings
Enabled when: bEntityFocus and nGloObjFunction = CUT
Checked when:
Menu Actions
If nGloObjType[nGloCell[nGloAbsPos]] = ENTITY_Rule
Else
Set nGloDetailsObject = nGloCell[nGloAbsPos]
Set bGloDetailsOK = TRUE
Loop
If SalModalDialog(dlgDetails, hWndForm)
Set bGloDetailsOK = SalModalDialog(dlgDetailsCheck, hWndForm)
If bGloDetailsOK
Break

- 103 -
If SaIModalDialog(dlgRules, hWndForm)
    Set bGloChanged = TRUE
Else
    Call SaIMessageBox("Rules may only be defined for regular entities.", APPNAME, MB_OK(MB_IconAlertMsg))
Popup Menu & Select
Enabled when: bGloItemsVisible and bGloDetailsOK
Menu Item: 1
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity1)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity1)
Menu Item: 2
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity2)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity2)
Menu Item: 3
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity3)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity3)
Menu Item: 4
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity4)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity4)
Menu Separator
Menu Item: 5
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity5)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity5)
Menu Item: 6
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity6)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity6)
Menu Separator
Menu Item: 7
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity7)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity7)
Menu Item: 8
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity8)
    Checked when:
    Menu Actions
    Call SaISetFocus(strEntity8)
Menu Item: 9
    Keyboard Accelerator: (none)
    Menu Settings
    Enabled when: SalisWindowVisible(strEntity9)
Checked when:
Menu Actions
Call SalSelFocus(strEntity9)

Menu Item: 10
Keyboard Accelerator: (none)
Menu Settings
Enabled when: SalIsWindowVisible(strEntity10)
Checked when:
Menu Actions
Call SalSelFocus(strEntity10)

Menu Separator

Menu Item: &Print Screen
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalWaitCursor(TRUE)
Call SalPrintForm(hWndForm)
Call SalWaitCursor(FALSE)

Popup Menu: &Help
Enabled when:
Menu Item: &Contents
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalWinHelp(hWndForm,'osamhelp.hlp',HELP_Context,1,NULL)

Menu Item: &Glossary of Terms
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalWinHelp(hWndForm,'osamhelp.hlp',HELP_Context,14,NULL)

Menu Item: &How To Use Help
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalWinHelp(hWndForm,'winhelp.hlp',HELP_HelpOnHelp,0,NULL)

Menu Separator
Menu Item: &About OSAM* Designer...
Keyboard Accelerator: (none)
Menu Settings
Enabled when:
Checked when:
Menu Actions
Call SalModelDialog(dlgAbout,hWndForm)

Contents

1 Entities

Data Field: strExtra
Data
  Maximum Data Length: Default
Data Type: String
Editable? No

Display Settings
Window Location and Size
  Left: Default
  Top: Default
  Width: 0.25"
  Height: 0.25"
Visible? No
Border? Yes
Justify: Left
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Data Field: strEntity1
Data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 0.288"
Top: 0.531"
Width: 1.025"
Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On SAM_SetFocus
Call SetEntityFocus(1,dom1)
On WM_CHAR
Return FALSE
On MSG_Gray
Call SetDomainGray(dom1)
On MSG_DrawLine
If SallNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen or
nGloAbsPos = (nGloScreen*10)+10
Call SaISendMsg(hWndItem,MSG_ShowLine,wParam,0)
Else
Call SaIColorSet(hWndItem,COLOR_IdxWindow,COLOR_Green)
On MSG_ShowLine
Set nTo = wParam
If nTo > 1
If nTo = 2
Call SaIShowWindow(hWndLine[LN_1To2])
If nTo = 3
If SallsWindowVisible(strEntity2)
Call SaIShowWindow(hWndLine[LN_1To3a])
Call SaIShowWindow(hWndLine[LN_1To3b])
Call SaIShowWindow(hWndLine[LN_1To3c])
Else
Call SaIShowWindow(hWndLine[LN_1To2])
Call SaIShowWindow(hWndLine[LN_2To3])
If nTo = 4
If SallsWindowVisible(strEntity2) or SallsWindowVisible(strEntity3)
Call SaIShowWindow(hWndLine[LN_1To4a])
Call SaIShowWindow(hWndLine[LN_1To4b])
Call SaIShowWindow(hWndLine[LN_1To4c])
Else
Call SaIShowWindow(hWndLine[LN_1To2])
Call SaIShowWindow(hWndLine[LN_2To3])
Call SaIShowWindow(hWndLine[LN_3To4])
If nTo = 5
Call SaIShowWindow(hWndLine[LN_1To5])
If nTo = 6
Call SaIShowWindow(hWndLine[LN_1To6])
If nTo = 7
If SallsWindowVisible(strEntity5)
Call SaIShowWindow(hWndLine[LN_1To7a])
Call SaIShowWindow(hWndLine[LN_1To7b])
Call SaIShowWindow(hWndLine[LN_1To7c])
Else
Call SaIShowWindow(hWndLine[LN_1To5])
Call SaIShowWindow(hWndLine[LN_STo7])
If nTo = 8
Call SaIShowWindow(hWndLine[LN_1To8])
If nTo = 9
Call SaIShowWindow(hWndLine[LN_1To9])
If nTo = 10
Call SaIShowWindow(hWndLine[LN_1To10])
On MSG_HideLine
If SaINumberTruncate(nGloAbsPos/10,8,0) = nGloScreen or
nGloAbsPos = (nGloScreen*10)+10
Set nTo = wParam
If nTo = 2
Call SaIHideWindow(hWndLine[LN_1To2])
Return TRUE
If nTo = 3
If SaIWindowVisible(strEntity2)
Call SaIHideWindow(hWndLine[LN_1To3a])
Call SaIHideWindow(hWndLine[LN_1To3b])
Call SaIHideWindow(hWndLine[LN_1To3c])
Else
Call SaIHideWindow(hWndLine[LN_1To2])
Call SaIHideWindow(hWndLine[LN_2To3])
Call SaIHideWindow(hWndLine[LN_3To4])
Return TRUE
If nTo = 4
If SaIWindowVisible(strEntity2) or SaIWindowVisible(strEntity3)
Call SaIHideWindow(hWndLine[LN_1To4a])
Call SaIHideWindow(hWndLine[LN_1To4b])
Call SaIHideWindow(hWndLine[LN_1To4c])
Else
Call SaIHideWindow(hWndLine[LN_1To2])
Call SaIHideWindow(hWndLine[LN_2To3])
Call SaIHideWindow(hWndLine[LN_3To4])
Return TRUE
If nTo = 5
Call SaIHideWindow(hWndLine[LN_1To5])
Return TRUE
If nTo = 6
Call SaIHideWindow(hWndLine[LN_1To6])
Return TRUE
If nTo = 7
If SaIWindowVisible(strEntity5)
Call SaIHideWindow(hWndLine[LN_1To7a])
Call SaIHideWindow(hWndLine[LN_1To7b])
Call SaIHideWindow(hWndLine[LN_1To7c])
Else
Call SaIHideWindow(hWndLine[LN_1To5])
Call SaIHideWindow(hWndLine[LN_STo7])
Return TRUE
If nTo = 8
Call SaIHideWindow(hWndLine[LN_1To8])
Return TRUE
If nTo = 9
Call SaIHideWindow(hWndLine[LN_1To9])
Return TRUE
If nTo = 10
Call SaIHideWindow(hWndLine[LN_1To10])
Return TRUE
Else
Call SaIColorSet(hWndItem,COLOR_IndexWindow,COLOR_Cyan)
On MSG_Hide
Call HideEntity(dom1)
Set strScreenObject[1] = NULL
On MSG_Show
Call ShowDomain(dom1)
Call SaISetWindowText(hWndItem,strScreenObject[1],12)
On MSG_Change
Call SalGetWindowText(hWndItem,strEntityText,8)
If strEntityText = strGIoOJDlnGloCell[nGloAbsPos]
Call SalSetWindowText(hWndItem,strO_ID)
On MSG_Reset
Call ResetEntity()
On MSG_Copy
If nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = nGloNextPos
If nHBar > 0
Set nGloCell[nGloAbsPos-7] = nGloNextPos
If nHBar > 0 and nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+11)] = nGloNextPos
On MSG_Delete
If nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = 0
If nHBar > 0
Set nGloCell[nGloAbsPos-7] = 0
If nHBar > 1 and nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+11)] = 0
Data Field: strEntity2
Date
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 1.708"
Top: 0.531"
Width: 1.025"
Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On MSG_DrawLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
or nGloAbsPos = (nGloScreen*10)+10
Call SalSendMsg(hWndItem,MSG_ShowLine,wParam,0)
Else
Call SalColorSet(hWndItem, COLOR_IndexWindow, COLOR_Green)
On MSG_ShowLine
Set nTo = wParam
If nTo > 2
If nTo = 3
Call SalShowWindow(hWndLine[LN_2To3])
If nTo = 4
If SalShowWindowVisible(strEntity3)
Call SalShowWindow(hWndLine[LN_2To4a])
Call SalShowWindow(hWndLine[LN_2To4b])
Call SalShowWindow(hWndLine[LN_2To4c])
Else
Call SalShowWindow(hWndLine[LN_2To3])
Call SalShowWindow(hWndLine[LN_3To4])
Else
Call SalShowWindow(hWndLine[LN_2To3])
Call SalShowWindow(hWndLine[LN_3To4])
If nTo = 5
Call SalShowWindow(hWndLine[LN_2To5])
If nTo = 6
Call SalShowWindow(hWndLine[LN_2To6])
If nTo = 7
Call SalShowWindow(hWndLine[LN_2To7])
If nTo = 8
Call SalShowWindow(hWndLine[LN_2To8])
If nTo = 9
    Call SalShowWindow(hWndLine[LN_2To9])
If nTo = 10
    Call SalShowWindow(hWndLine[LN_2To10])

On MSG_HideLine
If salNumberTruncate(nGloAbsPos/10,0) = nGloScreen
    or nGloAbsPos = (nGloScreen*10)+10
    Set nTo = wParam
If nTo = 1
    Call SalHideWindow(hWndLine[LN_1To2])
    Return TRUE
If nTo = 3
    Call SalHideWindow(hWndLine[LN_2To3])
    Return TRUE
If nTo = 4
    If SalIsWindowVisible(strEntity3)
        Call SalHideWindow(hWndLine[LN_2To4a])
        Call SalHideWindow(hWndLine[LN_2To4b])
        Call SalHideWindow(hWndLine[LN_2To4c])
    Else
        Call SalHideWindow(hWndLine[LN_2To3])
        Call SalHideWindow(hWndLine[LN_3To4])
    Return TRUE
If nTo = 5
    Call SalHideWindow(hWndLine[LN_2To5])
    Return TRUE
If nTo = 6
    Call SalHideWindow(hWndLine[LN_2To6])
    Return TRUE
If nTo = 7
    Call SalHideWindow(hWndLine[LN_2To7])
    Return TRUE
If nTo = 8
    Call SalHideWindow(hWndLine[LN_2To8])
    Return TRUE
If nTo = 9
    Call SalHideWindow(hWndLine[LN_2To9])
    Return TRUE
If nTo = 10
    Call SalHideWindow(hWndLine[LN_2To10])
    Return TRUE
Else
    Call SalColorSet(hWndItem,COLOR_indexWindow,COLOR_Cyan)

On WM_CHAR
    Return FALSE
On SAM_SetFocus
    Call SetEntityFocus(2, dom2)
On MSG_Gray
    Call SetDomainGray(dom2)
On MSG_Hide
    Call HidDomain(dom2)
Set strScreenObject[2] = NULL
On MSG_Show
    Call ShowDomain(dom2)
    Call SalGetWindowText(hWndItem,strScreenObject[2],12)
On MSG_Change
    Call SalGetWindowText(hWndItem,strEntityText,8)
    If strEntityText = strGlo_ID[nGloCell[nGloAbsPos]]
        Call SalSetWindowText(hWndItem,strID)
On MSG_Reset
    Call ResetEntity()
Maximum Data Length: Default
Data Type: String
Editable? Yes

Display Settings
Window Location and Size
  Left: 3.288"
  Top: 0.531"
  Width: 1.025"
  Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan

Message Actions
On MSG_DrawLine
  If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen or nGloAbsPos = (nGloScreen*10)+10
    Call SalSendMsg(hWndItem,MSG_ShawLine,wParam,0)
Else
    Call SaIColorSet(hWndItem,COLOR_IndexWindow,COLOR_Green)
On MSG_ShowLine
  Set nTo = wParam
  If nTo > 3
    If nTo = 4
      Call SalShowWindow(hWndLine[LN_3To4])
    If nTo = 5
      Call SalShowWindow(hWndLine[LN_3To5])
    If nTo = 6
      Call SalShowWindow(hWndLine[LN_3To6])
    If nTo = 7
      Call SalShowWindow(hWndLine[LN_3To7])
    If nTo = 8
      Call SalShowWindow(hWndLine[LN_3To8])
    If nTo = 9
      Call SalShowWindow(hWndLine[LN_3To9])
    If nTo = 10
      Call SalShowWindow(hWndLine[LN_3To10])
On MSG_HideLine
  If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen or nGloAbsPos = (nGloScreen*10)+10
    Set nTo = wParam
    If nTo = 1
      If SalIsWindowVisible(strEntity2)
        Call SalHideWindow(hWndLine[LN_1To3a])
        Call SalHideWindow(hWndLine[LN_1To3b])
        Call SalHideWindow(hWndLine[LN_1To3c])
      Else
        Call SalHideWindow(hWndLine[LN_1To2])
        Call SalHideWindow(hWndLine[LN_2To3])
      Return TRUE
    If nTo = 2
      Call SalHideWindow(hWndLine[LN_2To3])
      Return TRUE
    If nTo = 4
      Call SalHideWindow(hWndLine[LN_3To4])
      Return TRUE
    If nTo = 5
      Call SalHideWindow(hWndLine[LN_3To5])
      Return TRUE
    If nTo = 6
      Call SalHideWindow(hWndLine[LN_3To6])
      Return TRUE
    If nTo = 7
      - 110 -
Call SahlHideWindow(hWndLine[LN_3To7])
Return TRUE
If nTo = 8
Call SahlHideWindow(hWndLine[LN_3To8])
Return TRUE
If nTo = 9
Call SahlHideWindow(hWndLine[LN_3To9])
Return TRUE
If nTo = 10
Call SahlHideWindow(hWndLine[LN_3To10])
Return TRUE
Else
Call SahlColorSet(hWndItem, COLOR_indexWindow, COLOR_Cyan)
End If
On WM_CHAR
Return FALSE
On SAM_SetFocus
Call SetEntityFocus(3, dom3)
End If
On MSG_Gray
Call SetDomainGray(dom3)
End If
On MSG_Hide
Call HideEntity(dom3)
Set strScreenObject[3] = NULL
End If
On MSG_Show
Call ShowDomain(dom3)
Call SahlGetWindowText(hWndItem, strScreenObject[3], 12)
End If
On MSG_Change
Call SahlGetWindowText(hWndItem, strEntityText, 8)
If strEntityText = strGloOJD[nGloCell[nGloAbsPos]]
Call SahlSetWindowText(hWndItem, strO_ID)
End If
On MSG_Reset
Call ResetEntity()
End If
On MSG_Copy
If nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = nGloNextPos
End If
On MSG_Delete
If nVBar > 0
Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = 0
Data Field: strEntity4
Data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 4.788"
Top: 0.531"
Width: 1.025"
Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On MSG_DrawLine
If SahlNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
or nGloAbsPos = (nGloScreen+1)*10
Call SahlSendMsg(hWndItem, MSG_ShowLine, wParam, 0)
Else
Call SahlColorSet(hWndItem, COLOR_indexWindow, COLOR_Green)
End If
On MSG_ShowLine
Set nTo = wParam
If nTo > 4
If nTo = 5

- 111 -
Call SaIShowWindow(hWndLine[LN_4To5])
If nTo = 6
    Call SaIShowWindow(hWndLine[LN_4To6])
If nTo = 7
    Call SaIShowWindow(hWndLine[LN_4To7])
If nTo = 8
    Call SaIShowWindow(hWndLine[LN_4To8])
If nTo = 9
    Call SaIShowWindow(hWndLine[LN_4To9])
If nTo = 10
    If SaIShowWindowVisible(strEntity6)
        Call SaIShowWindow(hWndLine[LN_4To10a])
        Call SaIShowWindow(hWndLine[LN_4To10b])
        Call SaIShowWindow(hWndLine[LN_4To10c])
    Else
        Call SaIShowWindow(hWndLine[LN_4To5])
        Call SaIShowWindow(hWndLine[LN_6To10])
    EndIf
EndIf

On MSG_HideLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
or nGloAbsPos = (nGloScreen*10)+10
    Set nTo = wParam
If nTo = 1
    If SaIShowWindowVisible(strEntity2) or SaIShowWindowVisible(strEntity3)
        Call SaIHideWindow(hWndLine[LN_1To4a])
        Call SaIHideWindow(hWndLine[LN_1To4b])
        Call SaIHideWindow(hWndLine[LN_1To4c])
    Else
        Call SaIHideWindow(hWndLine[LN_1To2])
        Call SaIHideWindow(hWndLine[LN_2To3])
        Call SaIHideWindow(hWndLine[LN_3To4])
    Return TRUE
EndIf
If nTo = 2
    If SaIShowWindowVisible(strEntity3)
        Call SaIHideWindow(hWndLine[LN_2To4a])
        Call SaIHideWindow(hWndLine[LN_2To4b])
        Call SaIHideWindow(hWndLine[LN_2To4c])
    Else
        Call SaIHideWindow(hWndLine[LN_2To3])
        Call SaIHideWindow(hWndLine[LN_3To4])
    Return TRUE
EndIf
If nTo = 3
    Call SaIHideWindow(hWndLine[LN_3To4])
    Return TRUE
If nTo = 5
    Call SaIHideWindow(hWndLine[LN_4To5])
    Return TRUE
If nTo = 6
    Call SaIHideWindow(hWndLine[LN_4To6])
    Return TRUE
If nTo = 7
    Call SaIHideWindow(hWndLine[LN_4To7])
    Return TRUE
If nTo = 8
    Call SaIHideWindow(hWndLine[LN_4To8])
    Return TRUE
If nTo = 9
    Call SaIHideWindow(hWndLine[LN_4To9])
    Return TRUE
If nTo = 10
    If SaIShowWindowVisible(strEntity6)
        Call SaIHideWindow(hWndLine[LN_4To10a])
        Call SaIHideWindow(hWndLine[LN_4To10b])
        Call SaIHideWindow(hWndLine[LN_4To10c])
    Else
        Call SaIHideWindow(hWndLine[LN_4To6])
        Call SaIHideWindow(hWndLine[LN_6To10])
    Return TRUE
EndIf
Else
    Call SaIColorSet(hWndItem,COLOR_IndexWindow,COLOR_Cyan)
EndIf

On WM_CHAR
  Return FALSE
On SAM_SetFocus
  Call SetEntityFocus(4,dom4)
On MSG_Gray
  Call SetDomainGray(dom4)
On MSG_Hide
  Call HideEntity(dom4)
  Set strScreenObject[4] = NULL
On MSG_Show
  Call ShowDomain(dom4)
  Call SetGetWindowText(hWndItem,strScreenObject[4],12)
On MSG_Change
  Call SetGetWindowText(hWndItem,strEntityText,8)
  If strEntityText = strGloO_ID[nGloCell[nGloAbsPos]]
    Call SetSetWindowText(hWndItem,strO_ID)
On MSG_Reset
  Call ResetEntity()
On MSG_Copy
  If nVBar > 0
    Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = nGloNextPos
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+7] = nGloNextPos
  If nHBar < MAX_SCROLL and nVBar > 0
    Set nGloCell[nGloAbsPos-(MAX_SCROLL*10-3)] = nGloNextPos
On MSG_Delete
  If nVBar > 0
    Set nGloCell[nGloAbsPos-(MAX_SCROLL*10+4)] = 0
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+7] = 0
  If nHBar < MAX_SCROLL and nVBar > 0
    Set nGloCell[nGloAbsPos-(MAX_SCROLL*10-3)] = 0

Data Field: strEntity5
  Data
    Max Data Length: Default
    Data Type: String
    Editable? Yes
  Display Settings
    Window Location and Size
      Left: 0.288"
      Top: 1.365"
      Width: 1.025"
      Height: 0.271"
    Visible? Yes
    Border? Yes
    Justify: Center
    Format: Unformatted
    Country: Default
    Font Name: MS Sans Serif
    Font Size: 8
    Font Enhancement: Default
    Text Color: Default
    Background Color: Cyan
  Message Actions
  On MSG_DrawLine
    If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
      or nGloAbsPos = (nGloScreen*10)+10
      Call SalSendMsg(hWndItem,MSG_ShowLine,wParam,0)
    Else
      Call SalColorSet(hWndItem,COLOR_IndexWindow,COLOR_Green)
  On MSG_ShowLine
    Set nTo = wParam
    If nTo > 5
      Call SalShowWindow(hWndLine[LN_5To6])
      If nTo = 6
      Call SalShowWindow(hWndLine[LN_5To7])
      If nTo = 7
      Call SalShowWindow(hWndLine[LN_5To8])
      If nTo = 8
      Call SalShowWindow(hWndLine[LN_5To9])
If nTo = 9
    Call SaIShowWindow(hWndUne[LN_5To9])
If nTo = 10
    Call SaIShowWindow(hWndUne[LN_5To10])

On MSG_HideLine
    If SaINumberTruncate(nGloAbsPos/10,6,0) = nGloScreen
        or nGloAbsPos = (nGloScreen*10)+10
        Set nTo = wParam
    If nTo = 1
        Call SaIHideWindow(hWndUne[LN_1To5])
        Return TRUE
    If nTo = 2
        Call SaIHideWindow(hWndUne[LN_2To5])
        Return TRUE
    If nTo = 3
        Call SaIHideWindow(hWndUne[LN_3To5])
        Return TRUE
    If nTo = 4
        Call SaIHideWindow(hWndUne[LN_4To5])
        Return TRUE
    If nTo = 6
        Call SaIHideWindow(hWndUne[LN_5To6])
        Return TRUE
    If nTo = 7
        Call SaIHideWindow(hWndUne[LN_5To7])
        Return TRUE
    If nTo = 8
        Call SaIHideWindow(hWndUne[LN_5To8])
        Return TRUE
    If nTo = 9
        Call SaIHideWindow(hWndUne[LN_5To9])
        Return TRUE
    If nTo = 10
        Call SaIHideWindow(hWndLine[LN_5To10])
        Return TRUE
Else
    Call SaIColorSet(hWndItem,COLOR-,ndexWindow,COLOR_Cyan)
On WM_CHAR
    Return FALSE
On SAM_SetFocus
    Call SelEntityFocus(5,dom5)
On MSG_Gray
    Call SetDomainGray(dom5)
    Call SaIGetWindowText(hWndItem,strScreenObject[5],12)
    If strEntityText = strGloO_ID[nGloCell[nGloAbsPos])
        Call SalSetWindowText(hWndItem,strO_ID)
On MSG_Reset
    Call ResetEntityO
Data
    strEntity6
    Data
    Maximum Data Length: Default
    Data Type: String
    Editable? Yes
Display Settings
    Window Location and Size
    Left: 4.788"
Top: 1.365"
Width: 1.025"
Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan

Message Actions
On MSG_DrawingLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
  or nGloAbsPos = (nGloScreen*10)+10
  Call SalSendMsg(hWndItem,MSG_ShowLine, wParam, 0)
Else
  Call SalColorSet(hWndItem, COLOR_IndexWindow, COLOR_Green)
On MSG_ShowLine
Set nTo = wParam
If nTo > 0
  If nTo = 7
    Call SalShowWindow(hWndLine[LN_6To7])
  If nTo = 8
    Call SalShowWindow(hWndLine[LN_6To8])
  If nTo = 9
    Call SalShowWindow(hWndLine[LN_6To9])
  If nTo = 10
    Call SalShowWindow(hWndLine[LN_6To10])
On MSG_HideLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
  or nGloAbsPos = (nGloScreen*10)+10
Set nTo = wParam
If nTo = 1
  Call SalHideWindow(hWndLine[LN_1To6])
  Return TRUE
If nTo = 2
  Call SalHideWindow(hWndLine[LN_2To6])
  Return TRUE
If nTo = 3
  Call SalHideWindow(hWndLine[LN_3To6])
  Return TRUE
If nTo = 4
  Call SalHideWindow(hWndLine[LN_4To6])
  Return TRUE
If nTo = 5
  Call SalHideWindow(hWndLine[LN_5To6])
  Return TRUE
If nTo = 6
  Call SalHideWindow(hWndLine[LN_6To7])
  Return TRUE
If nTo = 7
  Call SalHideWindow(hWndLine[LN_6To8])
  Return TRUE
If nTo = 8
  Call SalHideWindow(hWndLine[LN_6To9])
  Return TRUE
If nTo = 9
  Call SalHideWindow(hWndLine[LN_6To10])
  Return TRUE
Else
  Call SalColorSet(hWndItem, COLOR_IndexWindow, COLOR_Cyan)
On WM_CHAR
Return FALSE
On SAM_SELFocus
Call SelEntityFocus(6.dom6)
On MSG_Gray
Call SetDomainGray(dom6)
On MSG_Hide
  Call HideEntity(dom6)
  Set strScreenObject[6] = NULL
On MSG_Show
  Call ShowDomain(dom6)
  Call SaIGetWindowText(hWndItem, strScreenObject[6], 12)
On MSG_Change
  Call SaIGetWindowText(hWndItem, strEntityText, 8)
  If strEntityText = strGloOJD[nGloAbsPos] + nGlo_ID
    Call SaISetWindowText(hWndItem, strO_ID)
On MSG_Reset
  Call ResetEntity()
On MSG_Copy
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+9] = nGloNextPos
On MSG_Delete
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+9] = 0

Data Field: strEntity7

Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes

Display Settings
  Window Location and Size
    Left: 0.268"
    Top: 2.198"
    Width: 1.025"
    Height: 0.26"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: MS Sans Serif
  Font Size: 8
  Font Enhancement: Default
  Text Color: Default
  Background Color: Cyan

Message Actions
On MSG_DrawLine
  If SalNumberTruncate(nGloAbsPos/10, B, 0) = nGloScreen
    or nGloAbsPos = (nGloScreen*10)+10
    Call SaISendMsg(hWndItem, MSG_ShowLine, wParam, 0)
  Else
    Call SaIColorSet(hWndItem, COLOR_IndexWindow, COLOR_Green)
On MSG_ShowLine
  Set nTo = wParam
  If nTo = 7
    Call SaIShowWindow(hWndLine[LN_7To8])
  If nTo = 8
    Call SaIShowWindow(hWndLine[LN_7To9])
  If nTo = 9
    If SalWindowVisible(strEntity8)
      Call SaIShowWindow(hWndLine[LN_7To9])
    Call SaIShowWindow(hWndLine[LN_7To8])
    Call SaIShowWindow(hWndLine[LN_7To9])
    Else
      Call SaIShowWindow(hWndLine[LN_7To9])
    Call SaIShowWindow(hWndLine[LN_7To8])
    Call SaIShowWindow(hWndLine[LN_8To9])
  If nTo = 10
    If SalWindowVisible(strEntity8) or SalWindowVisible(strEntity9)
      Call SaIShowWindow(hWndLine[LN_7To10])
    Call SaIShowWindow(hWndLine[LN_7To9])
    Call SaIShowWindow(hWndLine[LN_7To10a])
    Call SaIShowWindow(hWndLine[LN_7To9a])
    Else
      Call SaIShowWindow(hWndLine[LN_7To10])
    Call SaIShowWindow(hWndLine[LN_8To9])
    Call SaIShowWindow(hWndLine[LN_8To10])
    Call SaIShowWindow(hWndLine[LN_9To10])
On MSG_HideLine
If SalNumberTruncate(nGloAbsPos10,8,0) = nGloScreen
or nGloAbsPos = (nGloScreen*10)+10
Set nTo = wParam
If nTo = 1
Call SalHideWindow(hWndLine[LN_1To7a])
Call SalHideWindow(hWndLine[LN_1To7b])
Call SalHideWindow(hWndLine[LN_1To7c])
Return TRUE
If nTo = 2
Call SalHideWindow(hWndLine[LN_2To7])
Return TRUE
If nTo = 3
Call SalHideWindow(hWndLine[LN_3To7])
Return TRUE
If nTo = 4
Call SalHideWindow(hWndLine[LN_4To7])
Return TRUE
If nTo = 5
Call SalHideWindow(hWndLine[LN_5To7])
Return TRUE
If nTo = 6
Call SalHideWindow(hWndLine[LN_6To7])
Return TRUE
If nTo = 7
Call SalHideWindow(hWndLine[LN_7To8])
Return TRUE
If nTo = 8
Call SalHideWindow(hWndLine[LN_7To8])
Return TRUE
If nTo = 9
If SalisWindowVisible(strEntity8)
Call SalHideWindow(hWndLine[LN_7To9a])
Call SalHideWindow(hWndLine[LN_7To9b])
Call SalHideWindow(hWndLine[LN_7To9c])
Else
Call SalHideWindow(hWndLine[LN_7To9])
Call SalHideWindow(hWndLine[LN_8To9])
Return TRUE
If nTo = 10
If SalisWindowVisible(strEntity8) or SalisWindowVisible(strEntity9)
Call SalHideWindow(hWndLine[LN_7To10a])
Call SalHideWindow(hWndLine[LN_7To10b])
Call SalHideWindow(hWndLine[LN_7To10c])
Else
Call SalHideWindow(hWndLine[LN_7To10])
Call SalHideWindow(hWndLine[LN_8To10])
Call SalHideWindow(hWndLine[LN_9To10])
Return TRUE
Else
Call SalColorSet(hWndItem,COLOR_IndentWindow,COLOR_Cyan)
On WM_CHAR
Return FALSE
On SAM_SetFocus
Call SetEntityFocus(dom7)
On MSG_Gray
Call SelDomainGray(dom7)
On MSG_Hide
Call HideEntity(dom7)
Set strScreenObject[7] = NULL
On MSG_Show
Call ShowDomain(dom7)
Call SalGetWindowText(hWndItem,strScreenObject[7],12)
On MSG_Change
Call SalGetWindowText(hWndItem,strEntityText,6)
If strEntityText = strGloO_ID[nGloCell[nGloAbsPos]]
Call SalSetWindowText(hWndItem,strO_ID)
On MSG_Reset
Call ResetEntity()
On MSG_Copy
If nVBar < MAX_SCROLL
Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = nGloNextPos
If nHBar > 0
  Set nGloCell[nGloAbsPos-7] = nGloNextPos
If nHBar > 0 and nVBar < MAX_SCROLL
  Set nGloCell[nGloAbsPos+(MAX_SCROLL*10-3)] = nGloNextPos
On MSG_Delete
  If nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = 0
  If nHBar > 0
    Set nGloCell[nGloAbsPos-7] = 0
  If nHBar > 0 and nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = 0

Data Field: strEntity9

Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes

Display Settings
  Window Location and Size
    Left: 1.788"
    Top: 2.198"
    Width: 1.025"
    Height: 0.26"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: MS Sans Serif
  Font Size: 8
  Font Enhancement: Default
  Text Color: Default
  Background Color: Cyan

Message Actions
  On MSG_DrawLine
    If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
      or nGloAbsPos = (nGloScreen*10)+10
      Call SalSendMsg(hWndItem,MSG_ShowLine,wParam,0)
    Else
      Call SalColorSet(hWndItem,COLOR/IndexWindow,COLOR_Green)
  On MSG_ShowLine
    Set nTo = wParam
    If nTo > 8
      Call SalShowWindow(hWndItem[LN_8To9])
    If nTo = 9
      Call SalShowWindow(hWndItem[LN_9To10])
    If nTo = 10
      If SalWindowVisible(strEntity9)
        Call SalShowWindow(hWndItem[LN_10To11])
      Call SalShowWindow(hWndItem[LN_10To11])
      Call SalShowWindow(hWndItem[LN_11To12])
    Else
      Call SalShowWindow(hWndItem[LN_8To9])
      Call SalShowWindow(hWndItem[LN_9To10])
  On MSG_HideLine
    If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
      or nGloAbsPos = (nGloScreen*10)+10
      Set nTo = wParam
      If nTo = 1
        Call SalHideWindow(hWndItem[LN_1To2])
        Return TRUE
      If nTo = 2
        Call SalHideWindow(hWndItem[LN_2To3])
        Return TRUE
      If nTo = 3
        Call SalHideWindow(hWndItem[LN_3To4])
        Return TRUE
      If nTo = 4
        Call SalHideWindow(hWndItem[LN_4To5])
        Return TRUE
      If nTo = 5
Call SalHldeWindow(hWndUne[LN_5To9])
Return TRUE
If nTo = 6
Call SalHldeWindow(hWndUne[LN_6To9])
Return TRUE
If nTo = 7
Call SalHldeWindow(hWndUne[LN_7To9])
Return TRUE
If nTo = 9
Call SalHldeWindow(hWndUne[LN_8To9])
Return TRUE
If nTo = 10
If SalisWindowVisible(strEntity9)
Call SalHldeWindow(hWndUne[LN_8To10a])
Call SalHldeWindow(hWndUne[LN_8To10b])
Call SalHldeWindow(hWndUne[LN_8To10c])
Else
Call SalHldeWindow(hWndUne[LN_9To9])
Call SalHldeWindow(hWndUne[LN_9To10])
Return TRUE
Else
Call SalColorSel(hWndItem,COLOR_LndexWindow,COLOR_Cyan)
On WM_CHAR
Return FALSE
On SAM_SetFocus
Call SalEntityFocus(B,domB)
On MSG_Gray
Call SalDomainGray(domB)
On MSG_Hide
Call HideEntity(domB)
Set strScreenObject[B] = NULL.
On MSG_Show
Call ShowDomain(domB)
Call SalItemsWindow(hWndItem,strScreenObject[B],12)
On MSG_Change
Call SalGetWindowText(hWndItem,strEntityText,B)
If strEntityText = strGloO_ID[nGloCell[nGloAbsPos]]
Call SalSetWindowText(hWndItem,strO_ID)
On MSG_Reset
Call ResetEntity()
On MSG_Copy
If nVBar < MAX_SCROLL
Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = nGloNextPos
On MSG_Delete
If nVBar < MAX_SCROLL
Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = 0
Data Field: strEntity9
Data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 3.288"
Top: 2.19B"
Width: 1.025" Height: 0.26"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On MSG_DrawLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
  or nGloAbsPos = (nGloScreen*10)+10
  Call SalSendMsg(hWndItem,MSG_ShowLine,wParam,0)
Else
  Call SalColorSel(hWndItem,COLOR_IndexWindow,COLOR_Green)
On MSG_ShowLine
Set nTo = wParam
If nTo = 10
  Call SalShowWindow(hWndItem[LN_9To10])
On MSG_HideLine
If SalNumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
  or nGloAbsPos = (nGloScreen*10)+10
  Set nTo = wParam
If nTo = 1
  Call SalHideWindow(hWndItem[LN_1To9])
  Return TRUE
If nTo = 2
  Call SalHideWindow(hWndItem[LN_2To9])
  Return TRUE
If nTo = 3
  Call SalHideWindow(hWndItem[LN_3To9])
  Return TRUE
If nTo = 4
  Call SalHideWindow(hWndItem[LN_4To9])
  Return TRUE
If nTo = 5
  Call SalHideWindow(hWndItem[LN_5To9])
  Return TRUE
If nTo = 6
  Call SalHideWindow(hWndItem[LN_6To9])
  Return TRUE
If nTo = 7
  If SalWindowVisible(strEntityB)
    Call SalHideWindow(hWndItem[LN_7To9])
    Call SalHideWindow(hWndItem[LN_7To9b])
    Call SalHideWindow(hWndItem[LN_7To9c])
  Else
    Call SalHideWindow(hWndItem[LN_7To9])
    Call SalHideWindow(hWndItem[LN_7To9b])
    Call SalHideWindow(hWndItem[LN_7To9c])
  Return TRUE
If nTo = 8
  Call SalHideWindow(hWndItem[LN_8To9])
  Return TRUE
If nTo = 9
  Call SalHideWindow(hWndItem[LN_9To10])
  Return TRUE
Else
  Call SalColorSel(hWndItem,COLOR_IndexWindow,COLOR_Cyan)
On WM_CHAR
Return FALSE
On SAL_SelFocus
Call SetEntityFocus(9,dom9)
On MSG_Gray
Call SelDomainGray(dom9)
On MSG_Hide
Call HideEntity(dom9)
Set strScreenObject[9] = NULL
On MSG_Show
Call ShowDomain(dom9)
Call SalGetWindowText(hWndItem,strScreenObject[9],12)
On MSG_Change
Call SalGetWindowText(hWndItem,strEntityText,8)
If strEntityText = strGloO_(nGloAbsPos)
  Call SalSetWindowTitle(hWndItem,strGloO_)}
On MSG_Reset
Call ReselEntity()
On MSG_Copy
If nVBar < MAX_SCROLL
  Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = nGloNextPos
- 120 -
On MSG_Delete
If nVBar < MAX_SCROLL
  Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = 0

Data Field: strEntity10
Data
  Maximum Data Length: Default
  Data Type: String
  Editable?: Yes

Display Settings
Window Location and Size
  Left: 4.788"
  Top: 2.198"
  Width: 1.025"
  Height: 0.26"
  Visible?: Yes
  Border?: Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: MS Sans Serif
  Font Size: 8
  Font Enhancement: Default
  Text Color: Default
  Background Color: Cyan

Message Actions
On WM_CHAR
  Return FALSE

On MSG_DrawLine
  If SaINumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
    or nGloAbsPos = (nGloScreen*10)+10
  Else
    Call SaIColorSet(hWndItem,COLOR_IndexWindow,COLOR_Green)

On SAM_SetFocus
  Call SaEntityFocus(10,dom10)

On MSG_Gray
  Call SetDomainGray(dom10)

On MSG_Hide
  Call HideEntity(dom10)
  Set strScreenObject[10] = NULL

On MSG_Show
  Call ShowDomain(dom10)
  Call SaIGetWindowText(hWndItem,strScreenObject[10],12)

On MSG_Change
  Call SaIGetWindowText(hWndItem,strEntityText,8)
  If strEntityText = strGloO_IO[nGloCell[nGloAbsPos]]
    Call SaIGetWindowText(hWndItem,strO_IO)

On MSG_Reset
  Call ResetEntity()

On MSG_Copy
  If nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = nGloNextPos
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+7] = nGloNextPos
  If nHBar < MAX_SCROLL and nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+11)] = nGloNextPos

On MSG_Delete
  If nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+4)] = 0
  If nHBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+7] = 0
  If nHBar < MAX_SCROLL and nVBar < MAX_SCROLL
    Set nGloCell[nGloAbsPos+(MAX_SCROLL*10+11)] = 0

On MSG_HideLine
  If SaINumberTruncate(nGloAbsPos/10,8,0) = nGloScreen
    or nGloAbsPos = (nGloScreen*10)+10
    Set nTo = wParam
  If nTo = 1
    Call SalHideWindow(hWndItem[LN_1To10])
    Return TRUE
If nTo = 2
  Call sAllHideWindow(hWndLine[LN_2To10])
  Return TRUE
If nTo = 3
  Call sAllHideWindow(hWndLine[LN_3To10])
  Return TRUE
If nTo = 4
  If sAllWindowVisible(strEntity6)
    Call sAllHideWindow(hWndLine[LN_4To10a])
    Call sAllHideWindow(hWndLine[LN_4To10b])
    Call sAllHideWindow(hWndLine[LN_4To10c])
  Else
    Call sAllHideWindow(hWndLine[LN_4To10])
    Call sAllHideWindow(hWndLine[LN_6To10])
  Return TRUE
If nTo = 5
  Call sAllHideWindow(hWndLine[LN_5To10])
  Return TRUE
If nTo = 6
  Call sAllHideWindow(hWndLine[LN_6To10])
  Return TRUE
If nTo = 7
  If sAllWindowVisible(strEntity8) or sAllWindowVisible(strEntity9)
    Call sAllHideWindow(hWndLine[LN_7To10a])
    Call sAllHideWindow(hWndLine[LN_7To10b])
    Call sAllHideWindow(hWndLine[LN_7To10c])
  Else
    Call sAllHideWindow(hWndLine[LN_7To10])
    Call sAllHideWindow(hWndLine[LN_8To10])
    Call sAllHideWindow(hWndLine[LN_9To10])
  Return TRUE
If nTo = 8
  If sAllWindowVisible(strEntity9)
    Call sAllHideWindow(hWndLine[LN_8To10a])
    Call sAllHideWindow(hWndLine[LN_8To10b])
    Call sAllHideWindow(hWndLine[LN_8To10c])
  Else
    Call sAllHideWindow(hWndLine[LN_8To10])
    Call sAllHideWindow(hWndLine[LN_9To10])
  Return TRUE
If nTo = 9
  Call sAllHideWindow(hWndLine[LN_9To10])
  Return TRUE
Else
  Call sAllColorSet(hWndItem, COLOR_IndexWindow, COLOR_Cyan)

// Horizontal Scroll Bar: hBar
Window Location and Size
Left:  0.0"
Top:  2.979"
Width: 5.838"
Height: 0.198"
Visible? Yes
Message Actions
On SAM_Create
  Call sAllScrollSetRange(hWndItem,0,MAX_SCROLL,1,1)
On SAM_SelFocus
  Set bEntityFocus = FALSE
On SAM_ScrollBar
  If wParam = SB_ThumbTrack
    Call sAllScrollGetPos(hWndItem,nHBar)
    Set df_nHBar = nHBar-MAX_SCROLL/2
    Set nGloScreen = (nVBar*(MAX_SCROLL+1)+nHBar
    If nGloScreen != nPriorScreen
      Set nPriorScreen = nGloScreen
    Call sAllPosIMsg(pbOrigin,MSG_Redraw,O,O)
Data Field: df_nHBar
Data
  Maximum Data Length: 2
Data Type: Number
Editable? Yes

Display Settings
Window Location and Size
Left: 5.025"
Top: 2.979"
Width: 0.25"
Height: 0.198"
Visible? Yes
Border? Yes
Justify: Left

Format: Uppercase
Country: Default
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: Default
Text Color: Default
Background Color: White

Message Actions
On SAM_SetFocus
Set bEntityFocus = FALSE

On SAM_FieldEdit
If SalisNuII(hWndItem)
   Call SalisScrollGetPos(hWnd,nHBar)
   Set df_nHBar = nHBar*MAX_SCROLL/2
If df_nHBar < -MAX_SCROLL/2
   Set df_nHBar = -MAX_SCROLL/2
If df_nHBar > MAX_SCROLL/2
   Set df_nHBar = MAX_SCROLL/2
Set nHBar = df_nHBar+MAX_SCROLL/2
Call SalisScrollSetPos(hWnd,nHBar)
Set nGloScreen = (nVBar*(MAX_SCROLL+1))+nHBar
If nGloScreen = nPriorScreen
   Set nPriorScreen = nGloScreen
   Call SalisScrollSetRange(hWndItem,0,MAX_SCROLL,1,1)
   Call SalisScrollSetPos(hWnd,nVBar)
On WM_CHAR
   If wParam = 46
      Return FALSE

Vertical Scroll Bar: vBar
Window Location and Size
Left: 6.075"
Top: -0.01"
Width: 0.25"
Height: 2.813"
Visible? Yes

Message Actions
On SAM_Create
   Call SalisScrollSetRange(hWndItem,0,MAX_SCROLL,1,1)
On SAM_SetFocus
   Set bEntityFocus = FALSE
On SAM_ScrollBar
   If wParam = SB_ThumbTrack
      Call SalisScrollGetPos(hWndItem,nVBar)
      Set df_nVBar = nVBar*MAX_SCROLL/2
      Set nGloScreen = (nVBar*(MAX_SCROLL+1))+nHBar
      If nGloScreen = nPriorScreen
         Set nPriorScreen = nGloScreen
         Call SalisScrollSetRange(hWndItem,0,MAX_SCROLL,1,1)
         Call SalisScrollSetPos(hWnd,nVBar)
         Call SalisScrollSetRange(hWndItem,0,MAX_SCROLL,1,1)

Data Field: df_nVBar

Data
Maximum Data Length: 2
Data Type: Number
Editable? Yes

Display Settings
Window Location and Size
Left: 6.075"
Top: -2.792"
Width: 0.25"
Height: 0.198"
On SAM_Create
    Call SalPostMsg(hWndItem,SAM_Click,0,0)
On MSG_Redraw
    Call Redraw()
On SAM_Click
    !
    ! Reset the workspace to screen(0,0)
    !
    Set df_nHBar = 0
    Set df_nVBar = 0
    Set nHBar = df_nHBar+MAX_SCROLL/2
    Set nVBar = df_nVBar+MAX_SCROLL/2
    Call SalScrollSelPos(hBar,nHBar)
    Call SalScrollSelPos(vBar,nVBar)
    Set nGloScreen = (nVBar*(MAX_SCROLL+1))\nHBar
    If not bStartApp
        Call SalPostMsg(hWndItem,MSG_Redraw,0,0)
    Else
        Set bStartApp = FALSE
    ! Extra Join Lines
Line
Coordinates
    Begin X: 3.388"
    Begin Y: 2.623"
    End X: 4.189"
End Y: 2.823"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 1.9"
Begin Y: 2.823"
End X: 2.7"
End Y: 2.823"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 3.37"
Begin Y: 0.156"
End X: 4.175"
End Y: 0.156"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 0.088"
Begin Y: 0.656"
End X: 0.688"
End Y: 0.656"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 5.388"
Begin Y: 0.656"
End X: 6.0"
End Y: 0.656"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 1.088"
Begin Y: 2.906"
End X: 4.988"
End Y: 2.906"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line Coordinates
Begin X: 1.088"
Begin Y: 0.073"
End X: 5.075"
End Y: 0.073"
Visible? Yes
Line Thickness: 1
Line Color: Default

Lines From Entity 1 (Top To Bottom, Left To Right) = 12
Begin Y: 1.49"
End X: 1.888"
End Y: 0.656"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line
Coordinates
Begin X: 2.588"
Begin Y: 0.656"
End X: 5.488"
End Y: 1.573"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line
Coordinates
Begin X: 0.788"
Begin Y: 2.24"
End X: 2.188"
End Y: 0.74"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line
Coordinates
Begin X: 2.388"
Begin Y: 0.656"
End X: 3.788"
End Y: 2.323"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line
Coordinates
Begin X: 2.588"
Begin Y: 0.656"
End X: 5.288"
End Y: 2.323"
Visible? Yes
Line Thickness: 1
Line Color: Default
Lines From Entity 3 (Top To Bottom, Left To Right) = 7
Line
Coordinates
Begin X: 3.288"
Begin Y: 0.646"
End X: 4.788"
End Y: 0.646"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line
Coordinates
Begin X: 0.638"
Begin Y: 1.531"
End X: 3.338"
End Y: 0.698"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 4.188"
Begin Y: 0.856"
End X: 5.288"
End Y: 1.406"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 0.788"
Begin Y: 2.323"
End X: 3.388"
End Y: 0.74"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 2.188"
Begin Y: 2.406"
End X: 3.688"
End Y: 0.74"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 3.786"
Begin Y: 0.74"
End X: 3.786"
End Y: 2.406"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 3.888"
Begin Y: 0.74"
End X: 5.388"
End Y: 2.406"
Visible? Yes
Line Thickness: 1
Line Color: Default

1 Lines From Entity 4 (Top To Bottom, Left To Right) = 7

Line
Coordinates
Begin X: 1.238"
Begin Y: 1.448"
End X: 5.436"
End Y: 0.615"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 5.385"
Begin Y: 0.74"
End X: 5.385"
End Y: 1.573"
Visible? Yes
Line Thickness: 1
Line Color: Default

Line
Coordinates
Begin X: 0.586"
Begin Y: 2.406"
End X: 5.388"
End Y: 0.573"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 2.288"
Begin Y: 2.406"
End X: 4.988"
End Y: 0.74"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 3.788"
Begin Y: 2.406"
End X: 5.088"
End Y: 0.74"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 5.988"
Begin Y: 0.656"
End X: 5.988"
End Y: 2.323"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 5.588"
Begin Y: 2.323"
End X: 5.988"
End Y: 2.323"
Visible? Yes
Line Thickness: 1
Line Color: Default
1 Lines From Entity 5 (Top To Bottom, Left To Right) = 5
Line Coordinates
Begin X: 1.188"
Begin Y: 1.521"
End X: 4.888"
End Y: 1.521"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 0.688"
Begin Y: 1.573"
End X: 0.688"
End Y: 2.323"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Coordinates
Begin X: 0.788"
Begin Y: 1.573"
End X: 1.688"
End Y: 2.24"
Visible? Yes
Line Thickness: 1
Line Color: Default
Line Color: Default
<table>
<thead>
<tr>
<th>Extra Line (First Child)</th>
</tr>
</thead>
</table>
| **Line**
| Coordinates
| Begin X: 0.588"
| Begin Y: 2.656"
| End X: 0.988"
| End Y: 2.656"
| Visible? Yes
| Line Thickness: 1
| Line Color: Default
| Total Of 62 Lines
| Domains
| Frame
| Window Location and Size
| Left: 5.75"
| Top: 3.55"
| Width: 0.543"
| Height: 0.44"
| Visible? Yes
| Corners: Square
| Border Style: Drop-Shadow
| Border Thickness: 2
| Border Color: Default
| Background Color: Default
| Picture: picUNFLogo
| Window Location and Size
| Left: 5.793"
| Top: 3.574"
| Width: 0.457"
| Height: 0.405"
| Visible? Yes
| File Name: unlogo.bmp
| Storage: Internal
| Fit: Scale
| Scaling
| Width: 100
| Height: 100
| Corners: Square
| Border Style: No Border
| Border Thickness: 1
| Tile To Parent? No
| Border Color: Default
| Background Color: Default
| Message Actions
| On SAM_Click
| Call SALModalDialog(dlgAbout,hWndForm)
| Picture: dom1
| Window Location and Size
| Left: 0.688"
| Top: 0.24"
| Width: 0.275"
| Height: 0.229"
| Visible? Yes
| File Name: circyan.bmp
| Storage: Internal
| Fit: Size to Fit
| Scaling
| Width: 100
| Height: 100
| Corners: Square
| Border Style: No Border
| Border Thickness: 1
| Tile To Parent? No
| Border Color: Black
| Background Color: Default
| Message Actions
| On SAM_Click
| Call SALSetFocus(strEntity1)
Visible? Yes
File Name: circyan.bmp
Storage: Internal
Fit: Size to Fit
Scaling:
  Width: 100
  Height: 100
Corners: Square
Border Style: No Border
Border Thickness: 1
Tie To Parent? No
Border Color: Black
Background Color: Default
Message Actions
  On SAM_Click
    Call SalSetFocus(strEntity8)
Picture: dom9
Window Location and Size
  Left: 3.688"
  Top: 2.49"
  Width: 0.275"
  Height: 0.229"
Visible? Yes
File Name: circyan.bmp
Storage: Internal
Fit: Size to Fit
Scaling:
  Width: 100
  Height: 100
Corners: Square
Border Style: No Border
Border Thickness: 1
Tie To Parent? No
Border Color: Black
Background Color: Default
Message Actions
  On SAM_Click
    Call SalSetFocus(strEntity9)
Picture: dom10
Window Location and Size
  Left: 5.188"
  Top: 2.5"
  Width: 0.275"
  Height: 0.219"
Visible? Yes
File Name: circyan.bmp
Storage: Internal
Fit: Size to Fit
Scaling:
  Width: 100
  Height: 100
Corners: Square
Border Style: No Border
Border Thickness: 1
Tie To Parent? No
Border Color: Black
Background Color: Default
Message Actions
  On SAM_Click
    Call SalSetFocus(strEntity10)
Background Text: N&ame:
Window Location and Size
  Left: 0.075"
  Top: 3.302"
  Width: 0.588"
  Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Message Actions
On MSG_Gray
    Set strName = strGloObjName[nGloCell[nGloAbsPos]]
    If bEntityFocus
        Call SaEnableWindow(hWndItem)
    Else
        Call SaDisableWindow(hWndItem)
    End If
On SAM_SetFocus
    Set bEntityFocus = FALSE
On SAM_FieldEdit
    ! Verify that the name is not used by any other object
    !
    If SaStrTrim(strName,strName) > 0 and strName != SPACE
        And strName != strGloObjName[nGloCell[nGloAbsPos]] and IsAlphaNumeric(strName)
        Call SaWaitCursor(TRUE)
    End If
    Set nGloCount = 1
    While nGlobObjectType[nGloCount] != 0
        If strName = strGloObjName[nGloCount]
            Call SaMessageBox("This Name Has Been Used. Please Create A New Name.", APPNAME,MB_OK|MB_|IconAsterisk)
            Call SaEnableWindow(hWndItem)
            Return FALSE
        End If
        Set nGloCount = nGloCount + 1
    End While
    If nGloCount = 0
        Set nGloCount = 10
    End If
    Call SaWaitCursor(FALSE)
    Return TRUE
Else
    Set strName = strGloObjName[nGloCell[nGloAbsPos]]
On MSG_Disable
    Call SaDisableWindow(hWndItem)
    Call SaClearField(hWndItem)
On SAM_Validate
    If IsAlphaNumeric(strName)
        Set bGloDetailOK = TRUE
        Call SaDrawMenuBar(frmObjMgr)
        Call SaPostMsg(hWndItem,SAM_FieldEdit,0,0)
        Return VALIDATE_OK
    Else
        Set bGloDetailOK = FALSE
        Call SaDrawMenuBar(frmObjMgr)
    End If
Return VALIDATE_Cancel

Background Text: &ID:
Window Location and Size
Left: 2.988"
Top: 3.281"
Width: 0.263"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strO_ID
Data
Maximum Data Length: 7
Date Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 3.288"
Top: 3.24"
Width: 0.938"
Height: 0.25"
Visible? Yes
Border? Yes
Justify: Left
Format: Uppercase
Country: Default
Font Name: MS Sans Serif
Font Size: 10
Font Enhancement: Default
Text Color: Default
Background Color: White
Message Actions
On MSG_Gray
Set strO_ID = strGloO_ID[nGloCell[nGloAbsPos]]
If bEntityFocus
Call SaIEnableWindow(hWndItem)
Else
Call SaIDisableWindow(hWndItem)
On SAM_SetFocus
Set bEntityFocus = FALSE
On SAM_FieldEdit
| Verify that the name entered is not used by any other object |
| If SaIStrTrim(strO_ID,strO_ID) > 0 and strO_ID != SPACE |
| and strO_ID != strGloO_ID[nGloCell[nGloAbsPos]] and IsAlphaNumeric(strO_ID) |
Call SaIWaitCursor(TRUE)
Set nGloCount = 1
While nGlobObjType[nGloCount] != 0
If strO_ID = strGloO_ID[nGloCount]
Call SaIMessageBox('This ID Has Been Used. Please Create A New ID', APPNAME,MB_OK|MB_iconAsterisk)
Call SaIWaitCursor(FALSE)
Return FALSE
Set nGloCount = nGloCount + 1
Set strGloO_ID[nGloCell[nGloAbsPos]] = strO_ID
Set nGloCount = SaINumberMod(nGloAbsPos,10)
If nGloCount = 0
Set nGloCount = 10
Call SaISetWindowText(hWndDF[nGloCount],strO_ID)
Call SaIWaitCursor(FALSE)
Else
Set strO_ID = strGloO_ID[nGloCell[nGloAbsPos]]
On SAM_Validate
If IsAlphaNumeric(strO_ID)
    Set bGloDetailOK = TRUE
    Call SaIDrawMenuBar(frmObjMgr)
    Call SaIPostMsg(hWndIem, SAM_FleldEdlt, 0, 0)
    Return VALIDATE_OK
Else
    Set bGloDetailOK = FALSE
    Call SaIDrawMenuBar(frmObjMgr)
    Return VALIDATE_Cancel
On MSG_Disable
    Call SaIDisableWindow(hWndIem)
    Call SaIClearField(hWndIem)
Data Field: strType
    Data
    Maximum Data Length: 35
    Data Type: String
    Editable? No
    Display Settings
    Window Location and Size
    Left: 4.275"
    Top: 3.271"
    Width: 2.125"
    Height: 0.25"
    Visible? Yes
    Border? No
    Justify: Left
    Format: Unformatted
    Country: Default
    Font Name: MS Sans Serif
    Font Size: 10
    Font Enhancement: Default
    Text Color: Black
    Background Color: Default
    Message Actions
    On MSG_Gray
    Select Case nGloObjType[nGloCell[nGloAbsPos]]
    Case ENTITY_Reregular
        Set strType = 'Type: Regular Entity'
    Break
    Case ENTITY_Composite
        Set strType = 'Type: Composite Entity'
    Break
    Case ENTITY_CrossProduct
        Set strType = 'Type: Cross Product Entity'
    Break
    Case DOMAIN
        Set strType = 'Type: Composite Domain'
    Break
    Default
        Set strType = NULL
    Background Text: Object &Note:
    Window Location and Size
    Left: 0.088"
    Top: 3.604"
    Width: 0.588"
    Height: 0.385"
    Visible? Yes
    Justify: Left
    Font Name: Default
    Font Size: Default
    Font Enhancement: Default
    Text Color: Default
    Background Color: Default
Multiline Field: strDescription
    Data
    Maximum Data Length: 254
    String Type: String
    Editable? Yes
    Display Settings
Data Field: strMessage

Data
Maximum Data Length: Default
Data Type: String
Editable? No

Display Settings
Window Location and Size
Left: 0.18"
Top: 4.05"
Width: 5.00"
Height: 0.25"
Visible? Yes
Border? No
Justify: Left
Format: Unformatted
Country: Default
Font Name: MS Sans Serif
Font Size: 6
Font Enhancement: None
Text Color: Default
Background Color: Default

Message Actions
Window Variables
Boolean: bEntityFocus
Boolean: bStartApp
Number: nHBar
Number: nPriorScreen
Number: nTemp
Number: nTo
Number: nVBar
String: strDevice
String: strDriver
String: strEntityText
String: strPort
String: strScreenObject[15]

Window handle Arrays

Window Handle: hWndDF[30]
Window Handle: hWndDom[15]
Window Handle: hWndLine[65]

Message Atrons
On SAM_Create
    Call SetWolCursor(TRUE)
    I get and save a window handle to the Form's menu bar
    Set hWndGloMenuBar = GetMenu(hWndForm)
    Set hWndGloSubMenu = GetSubMenu(hWndGloMenuBar, SELECT_MENU)
    I Load handles for data fields, pictures, and lines. Show certain fields, and initialize startup variables.
    Set bStartApp = TRUE
    Call SetHideWindow(hWndForm)
    Call LoadHandles(hWndForm,hWndLine,TYPE_Line)
    Call LoadHandles(hWndForm,hWndDF,TYPE_DataField)
    Call LoadHandles(hWndForm,hWndDom,TYPE_Picture)
    Set hWndDom[10] = hWndNULL
Call ShowWindow(strOJD)
Call ShowWindow(strName)
Call ShowWindow(strType)
Call ShowWindow(strMessage)
Call ShowWindow(pICUNFLogo)
Call ShowWindow(strNewBar)
Call ShowWindow(hWndGloMenuBar)
Call SendMsgToChildren(hWndForm,MSG_HIDE,0,0)
Call SendMsgToChildren(hWndForm,MSG_DISABLE,0,0)
Set nGloCount = 1

While nGloCount < 11
    Set nGloCount = nGloCount + 1
Set nGloScreen = 0
Set nGloObjFunction = DELETE
Set bGloChanged = FALSE
Set bGloDetailOK = TRUE
Set bGloInitialized = TRUE
Call PostMsg(hWndForm,MSG_CREATED,0,0)

On MSG_Created
    Refresh Select Menu, show main window, and open OSAM* file if one has been provided.
    Set bEntityFocus = FALSE
    Call ShowWindow(hWndForm)
    Call RefreshSelectMenu()
    Call DestroyWindow(hWndGloWindow)
    Call WotCursor(FALSE)
If strArgArray[1] != NULL
    Call ShowRight(strArgArray[1],4,strGloFileName)
    Call ShowUpper(strGloFileName, strGloFileName)
If strGloFileName = '.SAM'
    Set strGloFileName = strArgArray[1]
    Call OpenOSAMFile(strGloFileName)
Else
    Call MessageBox(strArgArray[1]," is not a valid OSAM* file.",APPNAME,MB_OK)
    Set strGloFileName = NULL

On SAM_Close
    I end the program
    I

- 141 -
If not FileContinue() Return FALSE

Dialog Box: dlgAbout
Title: About OSAM* Designer
Display Settings
Visible at Design time? No
Type of Dialog: Modal
Window Location and Size
Left: 0.45"
Top: 0.177"
Width: 5.243"
Height: 3.988"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Gray

Contents
Frame
Window Location and Size
Left: 0.186"
Top: 0.143"
Width: 0.543"
Height: 0.44"
Visible? Yes
Corners: Square
Border Style: Drop-Shadow
Border Thickness: 2
Border Color: Default
Background Color: Default
Picture: picUNFLogo
Window Location and Size
Left: 0.229"
Top: 0.167"
Width: 0.457"
Height: 0.405"
Visible? Yes
File Name: unfigo.bmp
Storage: Internal
Fit: Scale
Scaling
Width: 100
Height: 100
Corners: Square
Border Style: No Border
Border Thickness: 1
Tile To Parent? No
Border Color: Default
Background Color: Default
Message Actions
Pushbutton: pbOk
Title: Ok
Window Location and Size
Left: 4.029"
Top: 0.155"
Width: 0.957"
Height: 0.25"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Call CloseDialog(hWndForm,TRUE)

Pushbutton: pbCancel
Title: Ok
Window Location and Size
Call SAL::EndDialog(hWndForm, TRUE)

Background Text: OSAM* Deiligner
Window Location and Size
Left: 0.8"
Top: 0.179"
Width: 2.829"
Height: 0.19"
Visible? Yes
Justify: Left
Font Name: MS Sans Serif
Font Size: 10
Font Enhancement: Bold
Text Color: Default
Background Color: Default

Background Text: Copyright © 1992 by Paul F. Rabuck
Window Location and Size
Left: 0.844"
Top: 0.429"
Width: 3.057"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Background Text: With Special Thanks to:
Window Location and Size
Left: 0.2"
Top: 0.679"
Width: 2.729"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Background Text: Dr. Susan R. Wallace, Thesis Advisor and Committee Chairman
Window Location and Size
Left: 0.371"
Top: 0.917"
Width: 3.466"
Height: 0.391"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Background Text: and the Thesis Committee:
Window Location and Size
Left: 0.2"
Top: 1.333"
Width: 2.443"
Message Actions
On SAM_Create
Set mSubmit = "This software is submitted to the University of North Florida College of Computer and Information Sciences in partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences."
On SAM_SelFocus
Call SALSelfFocus(pbOk)
Multiline Field: mlPortions
Data
Maximum Data Length: Default
String Type: String
Editable? No
Display Settings
Border? No
Word Wrap? Yes
Vertical Scroll? No
Window Location and Size
Left: 0.2"
Top: 2.833"
Width: 4.771"
Height: 0.75"
Visible? Yes
Font Name: MS Sans Serif
Font Size: 8
Font Enhancement: None
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
Set mlPortions = 'The ideas presented within this software and the term "OSAM" are based on prior work conducted by Dr. Stanley Y. W. Su at the Database Systems Research and Development Center, University of Florida.'
On SAM_Create
Call SalSetFocus(pbOk)
Window Variables
Message Actions
Dialogue Box: dlgAddObject
Title: Create An Object
Display Settings
Visible at Design time? No
Type of Dialog: Modal
Window Location and Size
Left: 1.95"
Top: 0.25"
Width: 5.071"
Height: 3.893"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Contents
Background Text: Object ID:
Window Location and Size
Left: 0.143"
Top: 0.167"
Width: 0.986"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strO_ID
Data
Maximum Data Length: 7
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
  Left: 1.343"
  Top: 0.119"
  Width: 1.129"
  Height: 0.25"
Visible? Yes
Border? Yes
Justify: Left
Format: Uppercase
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On MSG_Chck
  If strGloErrorMessage = NULL
    If SaIStrTrim(strO_ID,strO_ID) = 0 or strO_ID = SPACE
      Set strGloErrorMessage = 'An Object ID Must Be Provided.'
      Call SaISetFocus(hWndItem)
  End If
On SAM_FieldEdit
  If IsAlphaNumeric(strO_ID)
    Set strName = strO_ID
  Else
    Return _VALIDATE _Ok
  End If
Background Text: Object Name:
Window Location and Size
  Left: 0.129"
  Top: 0.5"
  Width: 1.157"
  Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strName
Data
Maximum Data Length: 18
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
  Left: 1.343"
  Top: 0.452"
  Width: 3.457"
  Height: 0.265"
Visible? Yes
Border? Yes
Justify: Left
Format: Uppercase
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On MSG_Check
  If strGloErrorMessage = NULL
    If SaIStrTrim(strName, strName) = 0 or strName = SPACE
      Set strGloErrorMessage = 'An Object Name Must Be Provided.'
      Call SaISetFocus(hWndItem)
  On SAM_Create
    Call SalUstSetSelect(hWndItem, 0)
    Background Text: Object &Note:
      Window Location and Size:
        Left: 0.129"
        Top: 0.821"
        Width: 1.143"
        Height: 0.167"
      Visible? Yes
      Justify: Left
      Font Name: Default
      Font Size: Default
      Font Enhancement: Default
      Text Color: Default
      Background Color: Default
      List Initialization:
        Text: Regular Entity
        Text: Composite Domain
        Text: Cross Product Entity
        Text: Composite Entity
      Message Actions:
        On SAM_Create
          Call SalUstSetSelect(hWndItem, 0)
  Background Text: Object &Type:
    Window Location and Size:
      Left: 0.129"
      Top: 0.821"
      Width: 1.143"
      Height: 0.167"
    Visible? Yes
    Justify: Left
    Font Name: Default
    Font Size: Default
    Font Enhancement: Default
    Text Color: Default
    Background Color: Default
  Combo Box: strOb|Type
    Window Location and Size:
      Left: 1.343"
      Top: 0.771"
      Width: 2.486"
      Height: 1.024"
    Visible? Yes
    Editable? No
    String Type: String
    Maximum Data Length: Default
    Sorted? No
    Always Show List? No
    Vertical Scroll? Yes
    Font Name: Default
    Font Size: Default
    Font Enhancement: Default
    Text Color: Default
    Background Color: Default
    List Initialization:
      Text: Regular Entity
      Text: Composite Domain
      Text: Cross Product Entity
      Text: Composite Entity
      Message Actions:
        On SAM_Create
          Call SalUstSetSelect(hWndItem, 0)
  Background Text: Object &Type:
    Window Location and Size:
      Left: 0.129"
      Top: 0.821"
      Width: 1.143"
      Height: 0.167"
    Visible? Yes
    Justify: Left
    Font Name: Default
    Font Size: Default
    Font Enhancement: Default
    Text Color: Default
    Background Color: Default
    Multiline Field: strDescription
    Data:
      Maximum Data Length: 254
      String Type: String
      Editable? Yes
      Display Settings
      Border? Yes
Word Wrap? Yes
Vertical Scroll? Yes
Window Location and Size
   Left: 0.1"
   Top: 1.321"
   Width: 4.7"
   Height: 0.583"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
   On SAM_Validate
      Call ReplaceChar(strDescription,DEL,SPACE)
      Call ReplaceChar(strDescription",""")
      Return VALIDATE_OK
Group Box: Screen Position
Window Location and Size
   Left: 0.1"
   Top: 1.94"
   Width: 3.371"
   Height: 1.512"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: scrpos1
Data
   Maximum Data Length: Default
   Data Type: String
   Editable? Yes
Display Settings
   Window Location and Size
      Left: 0.3"
      Top: 2.274"
      Width: 0.629"
      Height: 0.262"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
   On WM_CHAR
      Return FALSE
On SAM_Create
   If SetPositionValue(1,TRUE)
      Set bOpenPositions = TRUE
      Set nPos = SetPositionFocus()
On SAM_SetFocus
   Set nPos = SetPositionFocus()
On MSG_Gray
   Call SetPositionBlack()
Data Field: scrpos2
Data
   Maximum Data Length: Default
   Data Type: String
   Editable? Yes
Display Settings
   Window Location and Size
Data Field: scrpos3
data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 1.1"
Top: 2.274"
Width: 0.629"
Height: 0.262"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On WM_CHAR
Return FALSE
On SAM_Create
If SetPositionValue(2,TRUE)
    If not bOpenPositions
        Set nPos = SetPositionFocus()
        Set bOpenPositions = TRUE
    End If
On SAM_SetFocus
Set nPos = SetPositionFocus()
On MSG_Gray
Call SetPositionBlack()
Data Field: scrpos4
data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 1.1"
Top: 2.274"
Width: 0.629"
Height: 0.262"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On WM_CHAR
Return FALSE
On SAM_Create
If SetPositionValue(3,TRUE)
    If not bOpenPositions
        Set nPos = SetPositionFocus()
        Set bOpenPositions = TRUE
    End If
On SAM_SetFocus
Set nPos = SetPositionFocus()
On MSG_Gray
Call SetPositionBlack()
Message Actions
On WM_CHAR
Return FALSE
On SAM_Create
If SetPositionValue(4, TRUE)
  If not bOpenPositions
    Set nPos = SetPositionFocus()
  Set bOpenPositions = TRUE
On SAM_SetFocus
Set nPos = SetPositionFocus()
On MSG_Gray
Call SetPositionBlack()

Data Field: scrpos5
Data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
  Left: 0.3"
  Top: 2.69"
  Width: 0.629"
  Height: 0.262"
Visible? Yes
Border? Yes
Justify: Center
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On WM_CHAR
Return FALSE
On SAM_Create
If SetPositionValue(5, TRUE)
  If not bOpenPositions
    Set nPos = SetPositionFocus()
  Set bOpenPositions = TRUE
On SAM_SetFocus
Set nPos = SetPositionFocus()
On MSG_Gray
Call SetPositionBlack()

Data Field: scrpos6
Data
Maximum Data Length: Default
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
  Left: 2.7"
  Top: 2.69"
  Width: 0.629"
  Height: 0.262"
Visible? Yes
Border? Yes
Message Actions
On WM_CHAR
  Return FALSE
On SAM_Create
  If SetPositionValue(6, TRUE)
  If not bOpenPositions
    Set nPos = SetPositionFocus()
    Set bOpenPositions = TRUE
  On SAM_SetFocus
    Set nPos = SetPositionFocus()
  On MSG_Gray
    Call SetPositionBlack()
Data Field: scrpos7
Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes
Display Settings
  Window Location and Size
    Left: 0.3"
    Top: 3.107"
    Width: 0.629"
    Height: 0.262"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Cyan
Message Actions
On WM_CHAR
  Return FALSE
On SAM_Create
  If SetPositionValue(7, TRUE)
  If not bOpenPositions
    Set nPos = SetPositionFocus()
    Set bOpenPositions = TRUE
  On SAM_SetFocus
    Set nPos = SetPositionFocus()
  On MSG_Gray
    Call SetPositionBlack()
Data Field: scrpos8
Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes
Display Settings
  Window Location and Size
    Left: 1.1"
    Top: 3.107"
    Width: 0.629"
    Height: 0.262"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On WM_CHAR
  Return FALSE
On SAM_Create
  If SetPositionValue(9,TRUE)
    If not bOpenPositions
      Set nPos = SetPositionFocus()
      Set bOpenPositions = TRUE
    On SAM_SetFocus
      Set nPos = SetPositionFocus()
    On MSG_Gray
      Call SetPositionBlack()
Data Field: scrpos9
  Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes
Display Settings
  Window Location and Size
    Left: 1.9"
    Top: 3.107"
    Width: 0.629"
    Height: 0.262"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Cyan
Message Actions
On WM_CHAR
  Return FALSE
On SAM_Create
  If SetPositionValue(9,TRUE)
    If not bOpenPositions
      Set nPos = SetPositionFocus()
      Set bOpenPositions = TRUE
    On SAM_SetFocus
      Set nPos = SetPositionFocus()
    On MSG_Gray
      Call SetPositionBlack()
Data Field: scrpos10
  Data
  Maximum Data Length: Default
  Data Type: String
  Editable? Yes
Display Settings
  Window Location and Size
    Left: 2.7"
    Top: 3.107"
    Width: 0.629"
    Height: 0.262"
  Visible? Yes
  Border? Yes
  Justify: Center
  Format: Unformatted
  Country: Default
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
Text Color: Default
Background Color: Cyan
Message Actions
On WM_CHAR
Return FALSE
On SAM_Create
If SetPositionValue(10,TRUE)
If not bOpenPositions
Set nPos = SetPositionFocus()
Set bOpenPositions = TRUE
On SAM_SetFocus
Set nPos = SetPositionFocus()
On MSG_Gray
Call SetPositionBlack()
Ppushbutton: pbOk
Title: Ok
Window Location and Size
Left: 3.6"
Top: 2.024"
Width: 1.2"
Height: 0.298"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click

! Verify that there are no errors with the entered object, and update the global object arrays to
! hold the entry.
!
If NoErrorsFound()
Call SalWaitCursor(TRUE)
Set nGloCount = 1
Set nGloNextPos = 0
While nGloObjType[nGloCount] 1= 0
If strName = strGloObjName[nGloCount]
Call SalMessageBox('This Name Has Been Used. Please Create A New Name.',

APPNAME,MB_OK|MB_IconAsterisk)
Call SalSetFocus(strName)
Call SalWaitCursor(FALSE)
Return FALSE
If strO_ID = strGlo_ID[nGloCount]
Call SalMessageBox('This ID Has Been Used. Please Create A New ID',

APPNAME,MB_OK|MB_IconAsterisk)
Call SalSetFocus(strO_ID)
Call SalWaitCursor(FALSE)
Return FALSE
If nGloObjType[nGloCount] = DELETE and nGloNextPos = 0
Set nGloNextPos = nGloCount
Set nGloCount = nGloCount + 1
Set nGloScreenPos = nGloPos
Set nGloEditPos = (nGloScreen*10)+nPos
Set nGloEditPos = 0
If nGloNextPos = 0
Set nGloNextPos = nGloCount
Set strGloObjName[nGloNextPos] = strName
Set strGloObjO_ID[nGloNextPos] = strO_ID
Set strGloObjDesc[nGloNextPos] = strDescription
Set nGloObjType[nGloNextPos] = SalListQuerySelection(strObjType)+1
Set nGloObjCell[nGloNextPos] = nGloAbsPos
Set nGloObjHBar[nGloNextPos] = nHBar
Set nGloObjVBar[nGloNextPos] = nVBar
Set nGloObjAttrPtr[nGloNextPos] = 0
Set nGloCell[nGloAbsPos] = nGloNextPos
If not bGloAddObject
If nGloObjFunction = CUT
Set strGloObjAssociations[nGloNextPos] = strGloObjAssociations[0]
Set strGloObjCREl[nGloNextPos] = strGloObjCREl[0]
Set strGloObjXRel[nGloNextPos] = strGloObjXRel[0]
Call AddAssociations(nGloNextPos)
Set nGloObjAttrPtr[nGloNextPos] = nGloObjAttrPtr[0]
If nGloObjFunction = COPY
Call CopyDetails(0, FALSE)
Call RestoreDetails(nGloNextPos)
Set nGloObjInsert[nGloNextPos] = strGloObjInsert[0]
Set nGloObjUpdate[nGloNextPos] = strGloObjUpdate[0]
Set nGloObjDelete[nGloNextPos] = strGloObjDelete[0]
Call SetWindowText(frmObjMgr.hWndDF[nPos], strGloObjName[nGloNextPos])
If nGloObjType[nGloNextPos] = DOMAIN
Call SetColorSet(frmObjMgr.hWndDF[nPos], COLOR_IndexWindow, COLOR_Gray)
Else:
Call SetColorSet(frmObjMgr.hWndDF[nPos], COLOR_IndexWindow, COLOR_Cyan)
Call SendMsg(frmObjMgr.hWndDF[nPos], MSG_Copy, 0, 0)
Call SendMsg(frmObjMgr.hWndDF[nPos], MSG_Show, 0, 0)
Call SendMsg(hWndDF[nPos], SAM_SetFocus, 0, 0)
Call Redraw()
Call StartCursor(FALSE)
Set bGloAddObject = FALSE
Call EndDialog(hWndForm, FALSE)
Set bGloChanged = TRUE
Set nGloObjFunction = DELETE
Call EndDialog(hWndForm, FALSE)
Pushbutton: pbCancel
Title: Cancel
Window Location and Size
Left: 3.6"
Top: 2.357"
Width: 1.2"
Height: 0.298"
Visible? Yes
Keyboard Accelerator: Esc
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Set bGloAddObject = FALSE
Call EndDialog(hWndForm, FALSE)
Window Variables
Boolean: bOpenPositions
Number: nPos
Number: nRelObj
String: strRelation
String: strRelObj
Message Actions
On SAM_Create
Call StartCursor(TRUE)
Set bOpenPositions = FALSE
Call HideWindow(hWndForm)
Call SendMsg(hWndForm, MSG_Created, 0, 0)
On MSG_Created
Call StartCursor(FALSE)
If bOpenPositions
If not bGloAddObject
Call DisableWindow(strObjType)
If nGloObjFunction = CUT
Set strO_ID = strGloO_ID[0]
Set strName = strGloObjName[0]
Call DisableWindow(strO_ID)
Call DisableWindow(strName)
Call DisableWindow(strDescription)
Call SetWindowText(hWndForm, 'Paste An Object')
If nGloObjFunction = COPY
- 154 -
Call SaisSetWindowText(hWndForm,"Copy \\strGloO_ID[0]\" To:\")
Set strDescription = strGloObjDesc[0]
Call SaisListSetSelect(strObjType,nGloObjType[0]-1)
Call SaisShowWindow(hWndForm)
Else
  Call SaisMessageBox("This Screen Is Full. Please Scroll To Another Screen On Which Fewer Than 10 Objects Are Depicted.",APPNAME,MB_OK|MB_ICONASTERISK)
  Call SaisEndDialog(hWndForm,FALSE)
On SAM_Close
  Call SaisSendMsg(pbCancel,SAM_Click,0,0)
Dialog Box: dlgAssocateRegular
Title: Associate An Object
Display Settings
  Visible at Design time? No
  Type of Dialog: Modal
  Window Location and Size
    Left: 0.186"
    Top: 1.052"
    Width: 6.586"
    Height: 2.833"
  Absolute Screen Location? Yes
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Default
Contents
  Background Text: &Show:
    Window Location and Size
      Left: 0.114"
      Top: 0.107"
      Width: 0.543"
      Height: 0.167"
      Visible? Yes
      Justify: Left
      Font Name: Default
      Font Size: Default
      Font Enhancement: Default
      Text Color: Default
      Background Color: Default
    Combo Box: strObjType
      Window Location and Size
        Left: 0.757"
        Top: 0.083"
        Width: 2.271"
        Height: 1.179"
        Visible? Yes
        Editable? No
        String Type: String
        Maximum Data Length: Default
        Sorted? No
        Always Show List? No
        Vertical Scroll? Yes
        Font Name: Default
        Font Size: Default
        Font Enhancement: Default
        Text Color: Default
        Background Color: Default
        List Initialization:
        Message Actions
  On SAM_Create
    If nGloObjType[nGloCell[nGloAbsPos]] = DOMAIN
      Call SaisListAdd(hWndItem,"Regular Entities")
      Call SaisListAdd(hWndItem,"Cross Product Entities")
      Call SaisListAdd(hWndItem,"Composite Domains")
    Else
      Call SaisListAdd(hWndItem,"Regular Entities")
      Call SaisListSetSelect(hWndItem,0)
On SAM Click
  Call SaISendMsg(lbObject,MSG_Load,0,0)

Background Text: &Object:
Window Location and Size
  Left: 0.1"
  Top: 0.417"
  Width: 0.629"
  Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strAssocID
Data
  Maximum Data Length: Default
  Data Type: String
  Editable? No
Display Settings
Window Location and Size
  Left: 0.757"
  Top: 0.393"
  Width: 1.486"
  Height: 0.28"
Visible? Yes
Border? Yes
Justify: Left
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
List Box: lbObject
Window Location and Size
  Left: 0.871"
  Top: 0.631"
  Width: 1.371"
  Height: 1.726"
Visible? Yes
Multiple selection? No
Sorted? Yes
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
List Initialization
Message Actions
On SAM_Create
  Call SaIPostMsg(hWndItem,MSG_Load,0,0)
On MSG_Load
  Call SaILogClear(hWndItem)
  Set strAssocID = NULL
  If strObjType = 'Regular Entities'
    Set nObjType = ENTITY_Regular
  If strObjType = 'Composite Domains'
    Set nObjType = DOMAIN
  If strObjType = 'Cross Product Entities'
    Set nObjType = ENTITY_CrossProduct
  If strObjType = 'Composite Entities'
    Set nObjType = ENTITY_Composite
  Set nListCount = 1

- 156 -
Retrieve unassociated objects ONLY and place into combo box.

Call SalNumberToStr(nGloCell[nGloAbsPos],0,strObjLoc)
Call SalWaitCursor(FALSE)
While nGloObjType[nListCount] != 0
if SalStrScan(strGloObjAssoclations[nListCount],#'lIstrObjLoc) = -1
if strGloO_ID[nListCount] != frmObjMgr.strO_ID and
   strGloO_ID[nListCount] == NULL and nGloObjType[nListCount] = nObjType
   if not IsDependent(nListCount,nGloCell[nGloAbsPos]) and
      not IsAncestor(nListCount,nGloCell[nGloAbsPos])
   Call SalNumberToStr(nListCount,O,strListCount)
   Call SalStrLower(strGloO_ID[nListCount],strListCount)
   Call SalStrAdd(hWndltem,strListCount)
   Set nListCount = nListCount + 1
If SalListQueryCount(hWndltem) = 0
   Call SalDisableWindow(hWndltem)
   Call SalDisableWindow(cbAssociationType)
   Set cbAssociationType = NULL
   Set nAssociation1 = NULL
   Set strAssocID = 'none'
   Else
      Call SalEnableWindow(hWndltem)
      Call SalPosMsg(cbAssociationType,MSG_Load,O,0)
   On SAM_Click
   Call SalListQueryText(hWndltem,SalListQuerySelection(hWndltem),strObjectID)
   Call SalStrRight(strObjectID,8,strListCount)
   Call SalStrLeft(strObjectID,8,strObjectID)
   Call SalStrTrim(strObjectID,strObjectID)
   Call SalStrUpper(strListCount,strListCount)
   Set nListCount = SalStrToNumber(strListCount)
   Call SalStrUpper(strObjectID,strObjectID)
   Set strAssocID = strObjectID
   Call SalSendMsg(cbAssociationType,SAM_Click,O,0)

Background Text: &Association:
Window Location and Size
   Left: 2.471"
   Top: 0.417"
   Width: 1.043"
   Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Combo Box: cbAssociationType
Window Location and Size
   Left: 3.529"
   Top: 0.381"
   Width: 2.771"
   Height: 1.024"
Visible? Yes
Editable? No
String Type: String
Maximum Data Length: Default
Sorted? Yes
Always Show List? No
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Click
   If not SalIsNull(strAssocID)
      Call SalStrLeft(cbAssociationType,1,strAssocType)
If \( \text{strAssocType} = 'G' \) 
Set \( \text{miAssociation1} = 'A(n) \) list\( \text{ObjectID} \) is a kind of \( \text{frmObjMgr.strO_ID} \). All attributes belonging to the \( \text{frmObjMgr.strO_ID} \) are inherited by the \( \text{strObjectID} \). 

Set \( \text{miAssociation1} = \text{miAssociation1} \) if referential constraints are included, only predefined \( \text{frmObjMgr.strO_ID} \) which are not associated with any \( \text{strObjectID} \)'s may be deleted.

If \( \text{strAssocType} = 'A' \) 
Set \( \text{miAssociation1} = 'A(n) \) list\( \text{ObjectID} \) can be associated with a predefined \( \text{frmObjMgr.strO_ID} \). Key attributes used to identify the \( \text{frmObjMgr.strO_ID} \) are retained by the \( \text{strObjectID} \). 

Set \( \text{miAssociation1} = \text{miAssociation1} \) if referential constraints are included, any \( \text{strObjectID} \) having a predefined \( \text{frmObjMgr.strO_ID} \) will have their \( \text{frmObjMgr.strO_ID} \) nullified if that \( \text{frmObjMgr.strO_ID} \) is deleted.

If \( \text{strAssocType} = '0' \) 
Set \( \text{miAssociation1} = 'Every Instance of a(n) \) list\( \text{ObjectID} \) can have only one \( \text{strObjectID} \). Every \( \text{strObjectID} \) having a predefined \( \text{frmObjMgr.strO_ID} \) will have its \( \text{frmObjMgr.strO_ID} \) nullified if that \( \text{frmObjMgr.strO_ID} \) is deleted.

If \( \text{strAssocType} = 'M' \) 
Set \( \text{miAssociation1} = 'Every \) list\( \text{ObjectID} \) can have one or more \( \text{strObjectID} \)'s. Every \( \text{strObjectID} \)'s which are not associated with any \( \text{strObjectID} \)'s may be deleted.

If \( \text{strAssocType} = 'X' \) 
Set \( \text{miAssociation1} = 'The \) list\( \text{ObjectID} \) is partially defined by the \( \text{frmObjMgr.strO_ID} \). Key attributes used to identify the \( \text{frmObjMgr.strO_ID} \) are retained as part of the \( \text{strObjectID} \). Numeric attributes belonging to \( \text{strObjectID} \) can be summarized by \( \text{frmObjMgr.strO_ID} \).

If \( \text{strAssocType} = 'C' \) 
Set \( \text{miAssociation1} = 'The \) \text{frmObjMgr.strO_ID} \) is partially defined by the set of all \( \text{strObjectID} \)'s. Selected attributes of all \( \text{strObjectID} \)'s are summarized into \( \text{frmObjMgr.strO_ID} \).

On MSG_Load 
Call SAIListClear(hWndItem) 
Call SAIListClear(hWndItem) 
Set \( \text{miAssociation1} = \text{miAssociation1} \) if \( \text{nObjType} = \text{ENTITY_Composite} \) 
Call SAIListInsert(hWndItem, -1, 'C - Composition') 
If \( \text{nObjType} = \text{ENTITY_CrossProduct} \) 
Call SAIListInsert(hWndItem, -1, 'X - Cross Product') 
If \( \text{nObjType} = \text{ENTITY_CrossProduct} \) and \( \text{nGloObjType} = \text{ENTITY_Composite} \) 
If \( \text{nGloObjType} = \text{DOMAIN} \) 
Call SAIListInsert(hWndItem, -1, 'G - Generalization') 
Call SAIListInsert(hWndItem, -1, 'A - Aggregation') 
If \( \text{nGloObjType} = \text{ENTITY_Composite} \) 
Call SAIListInsert(hWndItem, -1, 'C - Composition') 
If \( \text{nGloObjType} = \text{ENTITY-Regular} \) 
If \( \text{nGloObjType} = \text{DOMAIN} \) 
Call SAIListInsert(hWndItem, -1, 'G - Generalization') 
Call SAIListInsert(hWndItem, -1, 'A - Aggregation') 
Call SAIListInsert(hWndItem, -1, 'O - Interaction (one to one)') 
Call SAIListInsert(hWndItem, -1, 'M - Interaction (one to many)') 
If SAIListQueryCount(hWndItem) = 1 
Call SAIListSetSelect(hWndItem) 
Call SAIListEnableWindow(hWndItem) 
Else 
Call SAIListEnableWindow(hWndItem) 

Group Box: Description 
Window Location and Size 
Left: 2.329" 
Top: 0.655" 
Width: 3.971" 
Height: 1.702" 
Visible? Yes 
Font Name: Default 
Font Size: Default 
Font Enhancement: Default 
Text Color: Default 
Background Color: Default
Multiline Field: miAssociation1
Data
Maximum Data Length: Default
Editable? No
Display Settings
Border? No
Word Wrap? Yes
Vertical Scroll? No
Window Location and Size
Left: 2.457"
Top: 0.94"
Width: 3.7"
Height: 1.298"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_SetFocus
Call SalSetFocus(pbOk)
Pushbutton: pbOk
Title: Ok
Window Location and Size
Left: 3.857"
Top: 0.071"
Width: 1.186"
Height: 0.25"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
| I Check For An Incomplete Association Entry | I
| If SalIsNull(strAssocID) | Call SalMessageBox("Please Select An Object From The Object List.", APPNAME, MB_OK(MB_iconAsterisk))
| Call SalSetFocus(IObject) | Return FALSE
| If strAssocID = '(none)' | Call SalMessageBox("There are no other defined \["strObjType\]| to associate with \["frmObjMgr.strO_ID]\].", APPNAME, MB_OK(MB_iconAsterisk))
| Call SalSetFocus(strObjType) | Return FALSE
| If SalisNull(cbAssociationType) | Call SalMessageBox("Please Select An Association Type. ", APPNAME, MB_OK(MB_iconAsterisk))
| Call SalSetFocus(cbAssociationType) | Return FALSE
| I Update the strGloObjAssociations[] for both objects | I
| Set strAncestorAssociations = strGloObjAssociations[nGloCell[nGloAbsPos]]
| Set strDependentAssociations = strGloObjAssociations[nListCount]
| Call SalStrLower(strAssocType,strAssocType)
| Set strGloObjAssociations[nGloCell[nGloAbsPos]] = strGloObjAssociations[nGloCell[nGloAbsPos]] \\
| strGloObjAssociations[nListCount]["#"][strListCount]["#"][strAssocType]
| Call SalStrUpper(strAssocType,strAssocType)
| Call SalNumberToStr(nGloCell[nGloAbsPos],0,strObjLoc)
| Set strGloObjAssociations[nListCount] = strGloObjAssociations[nListCount]["#"][strObjLoc]["#"][strAssocType]
| I Verify that the association is valid (no duplicate errors.)

- 159 -
I If NoDuplicateDependent(nGloCell[nGloAbsPos],strDupes)
Call SaISendWindow(hWndForm)
If NoDuplicateDependentAttributes(nGloCell[nGloAbsPos],strAssocType)
Set nGloCount = 1
While nGloCount < 10
If nGloCell[(nGloScreen*10)+nGloCount] = nListCount
Set nRelPos = SalNumberMod(nGloAbsPos,10)
If nRelPos = 0
Set nRelPos = 10
If nRelPos < nGloCount
Call SaISendMsg(frmObjMgr.hWndDF[nGloCount],MSG_DrawLine,nGloCount,0)
Else
Call SaISendMsg(frmObjMgr.hWndDF[nGloCount],MSG_DrawLine,nRelPos,0)
Set nGloCount = 11
Else
Set nGloCount = nGloCount + 1
]
I Repaint the workspace to reflect the new association.
I Call RefreshLabels()
Set bGloChanged = TRUE
Call SaIEndDialog(hWndForm,TRUE)
Else
I Upon finding an error, restore associations to what they were prior to entry.
I Set strGloObjAssociations[nGloCell[nGloAbsPos]] = strAncestorAssociations
Set strGloObjAssociations[nListCount] = strDependentAssociations
Call SaIEndDialog(hWndForm,FAIL)
Else
Set strGloObjAssociations[nGloCell[nGloAbsPos]] = strAncestorAssociations
I I Upon finding an error, restore associations to what they were prior to entry.
I Set strGloObjAssociations[nListCount] = strDependentAssociations
Call SaIMessageBox(frmObjMgr,strObjID,'may not be associated with '||strAssocID||' because the following dependent objects would have duplicate attributes: '||strDupes,
APPNAME,MB_OK,MB_ICONASTERISK)
Pushbutton: pbCancel
Title: Cancel
Window Location and Size
Left: 5.114"
Top: 0.071"
Width: 1.186"
Height: 0.25"
Visible? Yes
Keyboard Accelerator: Esc
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On S.A.M. Click
Call SaIEndDialog(hWndForm,FAIL)
Window Variables
Boolean: bCreated
Boolean: bObjectFound
Boolean: bRelated
Number: nDummy[*]
Number: nFocusOrigPos
Number: nListCount
Number: nObjAncestor[*]
Number: nObjDependent[*]
Number: nObjType
Number: nRelPos
String: strAncestorAssociations
String: strAssocType
String: strDependentAssociations
String: strDummy[*]
Message Actions
On SAM_Create
  Call SaWaitCursor(TRUE)
  Call SaISetWindowText(hWndForm,'Associate An Object With '||strObjMgr.strO_ID)
  Set bCreated = FALSE
On MSG_Created
  if not bObjectFound
    Call SaIEndDialog(hWndForm,FALSE)
  Else
    Call SaISendMsg(ilObject,SAM_Click,0,0)
    Call SaIShowWindow(hWndForm)
On SAM_Close
  Call SaISendMsg(pbCancel,SAM_Click,0,0)

Dialog Box:dlgBuildSQL
Title: Build SQL File
Display Settings
  Visible at Design time? No
  Type of Dialog: Modal
Window Location and Size
  Left: 0.553
  Top: 0.035
  Width: 7.043
  Height: 4.44
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Contents
Group Box: Text Format
Window Location and Size
  Left: 0.114
  Top: 0.071
  Width: 5.3
  Height: 1.476
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Radio Button: rbCommands
  Title: Uppercase &Commands Only: CREATE TABLE tablename...
Window Location and Size
  Left: 0.286
  Top: 0.262
  Width: 4.971
  Height: 0.25
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Radio Button: rbObjects
  Title: Uppercase &Objects Only: create table TABLENAME...
Window Location and Size
  Left: 0.286
  Top: 0.534
  Width: 4.9
  Height: 0.25
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Radio Button: rbProper
Title: All &Propercase: Create Table Tablename...
Window Location and Size
Left: 0.286"
Top: 0.762"
Width: 4.2"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Radio Button: rbUpper
Title: All &Uppercase: CREATE TABLE TABLENAME...
Window Location and Size
Left: 0.286"
Top: 1.012"
Width: 4.614"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Radio Button: rbLower
Title: All &Lowercase: create table tablename...
Window Location and Size
Left: 0.286"
Top: 1.25"
Width: 4.414"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Background Text: SHOW:
Window Location and Size
Left: 0.114"
Top: 1.702"
Width: 0.557"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Combo Box: strObjType
Window Location and Size
Left: 0.714"
Top: 1.667"
Width: 2.586"
Height: 1.488"
Visible? Yes
Editable? No
String Type: String
Maximum Data Length: Default
Sorted? No
Always Show List? No
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
List Initialization:
Text: Every Object
Text: Regular Entities
Text: Simple Domains
Message Actions
On SAM_Create
Call SaListSetSelect(hWndItem, ALL)
Call SaListInsert(hWndItem, 1, 'Dependent Upon ' || frmObjMgr.strO_ID)
Call SaListInsert(hWndItem, 1, 'Depended on by ' || frmObjMgr.strO_ID)
Call SaSendMsg(lbObjects, MSG_Load, 0, 0)
If nObjType = ALL
   Call SaShowWindow(hWndForm)
On SAM_Click
Set cbRestrict = FALSE
Call SaSendMsg(cbRestrict, SAM_Click, 0, 0)
Call SaSendMsg(lbObjects, MSG_Load, 0, 0)
Check Box: cbRestrict
Title: &Restrict...
Window Location and Size
Left: 0.229"
Top: 1.964"
Width: 1.271"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Click
If cbRestrict
   Call SaShowWindow(lbObjects)
   Call SaSetWindowText(hWndItem, '&Restrict To:"
   If SaListQuerySelection(strObjType) = 0
      Set mlRestrict = 'Composites and Cross Products not shown here.'
   Else
      Set mlRestrict = NULL
   Else
      Call SaHideWindow(lbObjects)
      Call SaSetWindowText(hWndItem, '&Restrict...
   Set mlRestrict = NULL
List Box: lbObjects
Window Location and Size
Left: 1.657"
Top: 1.988"
Width: 1.429"
Height: 2.036"
Visible? Yes
Multiple selection? Yes
Sorted? Yes
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
List Initialization
Message Actions
On MSG_Load
Call SalWaitCursor(TRUE)
Set nObjType = SalListItemQuerySelection(strObjType)
If nObjType > 2
    Set nObjType = nObjType + 2
Call SalListClear(hWndItem)
! Show objects of a certain type (if not a summary attribute.)
! If nObjType < DEPENDENT_UPON
    Set nGloCount = 1
While nGloObjType[nGloCount] = 0
    If nGloObjType = ALL or nGloObjType[nGloCount] = nObjType
        If nGloObjType[nGloCount] = ENTITY_Composite and
            nGloObjType[nGloCount] = ENTITY_CrossProduct
            Call SalNumberToStr(nGloCount,0,strObject)
            Set strObject = strGloID[nGloCount][TAB][strObject]
            Call SalStrLower(strObject,strObject)
            Call SalListAdd(hWndItem,strObject)
            Set nGloCount = nGloCount + 1
        Else
            If nObjType = DEPENDENT_ON_BY
                Call GetAllRelatedObjects(nGloCell[nGloAbsPos],FALSE,
                    nGloRelated,nGloRelatedLevel,nGloRelatedType,strGloObjRelation)
            Else
                Call GetAllRelatedObjects(nGloCell[nGloAbsPos],TRUE,
                    nGloRelated,nGloRelatedLevel,nGloRelatedType,strGloObjRelation)
            Call SalListClear(hWndItem)
            Set nGloCount = 1
While nGloRelated[nGloCount] = -1
    Call SalNumberToStr(nGloRelated[nGloCount],0,strObject)
    If nGloRelatedType[nGloCount] = ENTITY_Composite and
        nGloRelatedType[nGloCount] = ENTITY_CrossProduct
        Set strObject = strGloID[nGloRelated[nGloCount]][TAB][strObject]
        Call SalStrLower(strObject,strObject)
        Call SalListAdd(hWndItem,strObject)
        Set nGloCount = nGloCount + 1
    Else
        Call SalWaitCursor(FALSE)
        Call SalListRedraw(hWndItem,TRUE)
        If SalListItemQueryCount(hWndItem) = 0
            If nObjType = ALL
                Call SalMessageBox('No Objects Have Been Created.',APPNAME,MB_OK|MB_IconAsterisk)
            Set nObjType = -1
            Call SalPostMsg(pbCancel,SAM_Click,0,0)
            Return FALSE
        Call SalHideWindow(hWndItem)
    On SAM_DoubleClick
        Call SalSendMsg(pbOk,SAM_Click,0,0)
    On SAM_Click
        Call SalListQueryText(lbObjects,SalListItemQuerySelection(lbObjects),strObject)
        Call SalStrTokenize(strObject,TAB,TAB,strObjParm)
        Call SalWaitCursor(FALSE)
Group Box: SQL Definitions
Window Location and Size
    Left: 3.386"  
    Top: 1.595"  
    Width: 3.414"  
    Height: 0.524"  
    Visible? Yes  
    Font Name: Default  
    Font Size: Default  
    Font Enhancement: Default  
    Text Color: Default  
    Background Color: Default  
Radio Button: rbNew  
Title: &New Tables  
Window Location and Size
    Left: 3.629"  

Radio Button: rbOverwrite
Title: Overwrite Tables
Window Location and Size
Left: 5.043"
Top: 1.81"
Width: 1.643"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions

Check Box: cbRules
Title: Build Rules Table...
Window Location and Size
Left: 3.386"
Top: 2.167"
Width: 2.043"
Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Click
If cbRules
    Call SaIShowWindow(dfRulesTable)
    Call SaISetTextWindow(hWndItem,"Build Rules Table:"
    Call SaISetFocus(dfRulesTable)
Else
    Call SaIHideWindow(dfRulesTable)
    Call SaISetTextWindow(hWndItem,"Build Rules Table..."
Data Field: dfRulesTable
Data
Maximum Data Length: 18
Data Type: String
Editable? Yes
Display Settings
Window Location and Size
Left: 3.643"
Top: 2.429"
Width: 3.143"
Height: 0.25"
Visible? Yes
Border? Yes
Justify: Left
Format: Uppercase
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
Call SaiHideWindow(hWndItem)

On SAM_KillFocus
  If SallSNull(hWndItem)
    Set cbRules = FALSE
    Call SalSendMsg(cbRules,SAM_Click,0,0)
  Return VALIDATE_OK

On SAM_Validate
  ! Verify that an alphanumeric rules table has been declared.
  ! If SallSNull(hWndItem)
    Set cbRules = FALSE
    Call SalSendMsg(cbRules,SAM_Click,0,0)
  Return VALIDATE_OK

If IsAlphaNumeric(dfRulesTable)
  If SalStrTrim(dfRulesTable,dfRulesTable) > 0 and dfRulesTable != SPACE
    Call SaiWaitCursor(TRUE)
    Set nGloCount = 1
    While nGloObjType[nGloCount] != 0
      If dfRulesTable = strGloObjName[nGloCount]
        Call SalMessageBoxCThls
          Name Has Been Used. Please Create A New Name.
        APPNAME,MB_Ok(/MB_iconAsterisk)
      Call SaiWaitCursor(FALSE)
      Return VALIDATE_Cancel
    Return VALIDATE_OK
  Else
    Return VALIDATE_Cancel

Check Box: cbInclude
  Title: Include Comments...
  Window Location and Size
    Left: 3.386"
    Top: 2.726"
    Width: 2.671"
    Height: 0.25"
  Visible? Yes
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Default
  Message Actions
    On SAM_Click
      If cbInclude
        Call SaiShowWindow(dfDelimiter)
        Call SaiSetWindowText(hWndItem,'Include Comments delimited by:')
      Else
        Call SaiHideWindow(dfDelimiter)
        Call SaiSetWindowText(hWndItem,'&Include Comments...')
  Data Field: dfDelimiter
  Data
    Maximum Data Length: 3
    Data Type: String
    Editable? Yes

Display Settings
  Window Location and Size
    Left: 6.329"
    Top: 2.736"
    Width: 0.471"
    Height: 0.25"
  Visible? Yes
  Border? Yes
  Justify: Left
  Format: Unformatted
  Country: Default
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
Background Color: Default

Message Actions
On SAM_Create
   Call SaIHideWindow(hWndItem)
   Set dDelimiter = ";"

Check Box: cbReferential
Title: Include &Foreign and Primary Keys
Window Location and Size
   Left: 3.386"
   Top: 2.988"
   Width: 3.043"
   Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Olick
   Set bGloReferential = cbReferential
On SAM_Create
   Set cbReferential = TRUE
   Set bGloReferential = cbReferential

Check Box: cbStoreComposites
Title: &Store Composite Entities
Window Location and Size
   Left: 3.386"
   Top: 3.25"
   Width: 3.1"
   Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
   Set cbStoreComposites = TRUE

Check Box: cbStoreCrossProducts
Title: &Store Cross Product Entities
Window Location and Size
   Left: 3.386"
   Top: 3.512"
   Width: 2.757"
   Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
   Set cbStoreCrossProducts = TRUE

Check Box: cbCommit
Title: &Issue Commit When Completed
Window Location and Size
   Left: 3.386"
   Top: 3.774"
   Width: 3.186"
   Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
  Set cbCommit = TRUE
Pushbutton: pbOk
  Title: Ok
  Window Location and Size
    Left: 5.529"
    Top: 0.143"
    Width: 1.271"
    Height: 0.274"
  Visible? Yes
  Keyboard Accelerator: Enter
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
Message Actions
On SAM_Click
  I
  I Prompt the user for a SQL file name.
  I
  Call ResetFileNames(strGloFileName,strGloFilePath,'sql')
  Call SaIStrLeft(strGloFilePath,SaIStrScan(strGloFilePath,'.'),strGloFilePath)
  Set strGloFilePath = strGloFilePath\".sql\"
  Call SaIStrLeft(strGloFileName,SaIStrScan(strGloFileName,'.'),strGloFileName)
  Set strGloFileName = strGloFileName\".sql\"
  If DlgSaveAs(hWndForm,strGloDefPath,APPNAME\" - Build SQL File;sql\",SQL Files\".sql\",OFN\PATHMUSTEXIST|OFN\HIDEREADONLY|OFN\OVERWRITEPROMPT,strGloFilePath,strGloFileName
    I
    I If a file name has been provided, hide window and call BuildSQLFile()
    I
    Call SaIStrLeft(strGloFilePath,SaIStrScan(strGloFilePath,','),strGloFilePath)
    Set bGloUpperCommands = rbCommands
    Set bGloOverwrite = rbOverwrite
    Set bGloStoreComposites = cbStoreComposites
    Set bGloStoreCrossProducts = cbStoreCrossProducts
    Call BuildSQLFile(strGloFilePath,lbObjects)
    Call SaIEndDialog(hWndForm,TRUE)
    Call SaIStrLeft(strGloFilePath,SaIStrScan(strGloFilePath,','),strGloFilePath)
    Set strGloFilePath = strGloFilePath\".sam\"
    Call SaIStrLeft(strGloFileName,SaIStrScan(strGloFileName,','),strGloFileName)
    Set strGloFileName = strGloFileName\".sam\"
Pushbutton: pbCancel
  Title: Cancel
  Window Location and Size
    Left: 5.529"
    Top: 0.476"
    Width: 1.271"
    Height: 0.274"
  Visible? Yes
  Keyboard Accelerator: Esc
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
Message Actions
On SAM_Click
  Call SaIEndDialog(hWndForm,FALSE)
Multiline Field: mlRestrict
  Data
    Maximum Data Length: Default
    String Type: String
    Editable? No
  Display Settings
    Border? No
    Word Wrap? Yes
    Vertical Scroll? No
  Window Location and Size
    Left: 0.1"
    Top: 3.512"
The user never sees this window.

After hiding itself, dlgDetailCheck creates a dlgDetails dialog, which also hides itself, for each object. As instances of dlgDetails are created, they are traversed for duplicates. After traversing each object, this loop terminates, and dlgDetailCheck destroys itself and returns TRUE. If a duplicate is found, a message is displayed and dlgDetailChecks returns FALSE.

Set bGloDetailOK = FALSE.
If NoDuplicateDependentAttributes(nGloCell[nGloAbsPos], 'G')
   Set bGloChanged = TRUE
   Call SalEndDialog(hWndForm, TRUE)
Else
   If SalMessageBox(frmObjMgr.strObj childbirth cannot be saved as is because the "[strGloSavedAttr]" attribute in [strGloObj_ID][GloDetailObject]) would be duplicated. If this is not corrected, changes just made to [frmObjMgr.strObj childbirth] will be lost. Do you wish to correct the changes?
   ,APPNAME, MB_YesNoMBIconQuestion) = IDYES
   Set nGloDetailObject = nGloCell[nGloAbsPos]
   Call SalEndDialog(hWndForm, FALSE)
Else
   Call RestoreDetails(nGloCell[nGloAbsPos])
   Call SalEndDialog(hWndForm, TRUE)

Dialog Box: dlgDetails
Title: Display Settings
Visible at Design time? No
Type of Dialog: Modal
Window Location and Size
  Left: 0.163"
  Top: 0.479"
  Width: 8.8"
  Height: 4.464"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Contents
Table Window: tblDetails
  Title: Icon
  File: Display Settings
  Visible at Design time? Yes
  Automatically Created at Runtime? Yes
  Initial State: Normal
  Maximizable? No
  Minimizable? No
  System Menu? No
  Resizable? No
  Window Location and Size
    Left: 0.0"
    Top: 0.5"
    Width: 8.657"
    Height: 3.631"
  Visible? Yes
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Default

Memory Settings
  Maximum Rows In Memory: Default
  Discardable? Yes

Menu
Contents
Column: colName
  Title: Name
  Visible? Yes
  Editable? Yes
  Maximum Data Length: 16
  Data Type: String
  Justify: Left
  Width: 1.629"
  Format: Lowercase
  Country: Default

Message Actions
  On MSG_Highlight
    Call SltTblSelFocusCell(hWndForm,nAttributeTblRow,hWndItem,-1,-1)
  On SAM_SelFocus
    Set hWndColFocus = hWndItem
  On SAM_Vaidate
    If IsAlphaNumeric(colName)
      Return VALIDATE_OK
    Else
      Return VALIDATE_Cancel

Column: colFrom
  Title: From
  Visible? Yes
  Editable? No
  Maximum Data Length: Default
  Data Type: String

- 170 -
<table>
<thead>
<tr>
<th>Column: colTableName</th>
<th>Title: Table</th>
<th>Visible? No</th>
<th>Editable? No</th>
<th>Maximum Data Length: Default</th>
<th>Data Type: String</th>
<th>Justify: Left</th>
<th>Width: 1.0&quot;</th>
<th>Format: Unformatted</th>
<th>Country: Default</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Column: colAssocType</th>
<th>Title: Assoc</th>
<th>Visible? Yes</th>
<th>Editable? No</th>
<th>Maximum Data Length: Default</th>
<th>Data Type: String</th>
<th>Justify: Center</th>
<th>Width: 0.586&quot;</th>
<th>Format: Unformatted</th>
<th>Country: Default</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Column: colKey</th>
<th>Title: Key</th>
<th>Visible? Yes</th>
<th>Editable? Yes</th>
<th>Maximum Data Length: 3</th>
<th>Data Type: String</th>
<th>Justify: Left</th>
<th>Width: 0.457&quot;</th>
<th>Format: Unformatted</th>
<th>Country: Default</th>
</tr>
</thead>
</table>

On SAM_AnyEdit

- If not SallsNull(hWndItem)
  - If colKey = 'Y' or colKey = 'y'
    - Set colKey = 'Yes'
    - Set colRequired = 'Yes'
    - Set colIndexed = 'Yes'
    - Return TRUE
  - Else
    - Set colKey = 'No'
    - Return TRUE
- Call SaIClearField(hWndItem)

On SAM_KillFocus

- If colKey = 'Yes' and SallsNull(colType)
  - Call SallPostMsg(colType, SAM_SetFocus, 0, 0)
  - Call SallPostMsg(colType, MSG_Highlight, 0, 0)

On MSG_Highlight

- Call SallTblSetFocusCell(hWndForm, nAttributeTblRow, hWndItem, -1, -1)

On SAM_SetFocus

- Set hWndColFocus = hWndItem
If not SaliNull(hWndItem)
    If colRequired = 'Y' or colRequired = 'y' or colKey = 'Yes'
        Set colRequired = 'Yes'
        Return TRUE
    If colRequired = 'N' or colRequired = 'n'
        Set colRequired = 'No'
        Return TRUE
    Call SalClearField(hWndItem)
On MSG_Highlight
    Call SalTblSetFocusCell(hWndForm,nAttributeTblRow,hWndItem,-1,-1)
On SAM_SetFocus
    Set hWndColFocus = hWndItem
Column: colIndexed
    Title: Index
    Visible? Yes
    Editable? Yes
    Maximum Data Length: 3
    Data Type: String
    Justify: Center
    Width: 0.529
    Format: Unformatted
    Country: Default
    Message Actions
    On SAM_AnyEdit
        If not SalisNull(hWndItem)
            If colIndexed = 'Y' or colIndexed = 'y' or colKey = 'Yes'
                Set colIndexed = 'Yes'
                Return TRUE
            If colIndexed = 'N' or colIndexed = 'n'
                Set colIndexed = 'No'
                Return TRUE
            Call SalClearField(hWndItem)
On MSG_Highlight
    Call SalTblSetFocusCell(hWndForm,nAttributeTblRow,hWndItem,-1,-1)
On SAM_SetFocus
    Set hWndColFocus = hWndItem
Column: colType
    Title: Type (Press spacebar to show.)
    Visible? Yes
    Editable? Yes
    Maximum Data Length: 50
    Data Type: String
    Justify: Left
    Width: 2.857
    Format: Unformatted
    Country: Default
    Message Actions
    On SAM_AnyEdit
        If not SalisNull(hWndItem)
            Call SalStrUpper(coIType,strCheck)
            Call SalStrRight(strCheck,1,strCheck)
            Set nCheck = SalStrScan(TYPE_CHOICES,strCheck)
            If nCheck != -1
                Call SalStrReplace(strColType[nCheck],SalStrScan(strColType[nCheck],',')1,NULL,
                    coIType)
                Set colItemType = nCheck
            Else
                Call SalClearField(hWndItem)
                Call SalSendMsg(dlgDetails,MSG_Show,0,0)
                Call SalPostMsg(hWndItem,MSG_Reset,0,0)
On MSG_Highlight
    Call SalTblSetFocusCell(hWndForm,nAttributeTblRow,hWndItem,-1,-1)
On SAM_SetFocus
    Set hWndColFocus = hWndItem
On MSG_Reset
    Call GetTypeSpecifics()
<table>
<thead>
<tr>
<th>Column</th>
<th>Title</th>
<th>Visible?</th>
<th>Editable?</th>
<th>Maximum Data Length</th>
<th>Data Type</th>
<th>Justify</th>
<th>Width</th>
<th>Format</th>
<th>Country</th>
<th>Message Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>colCRel</td>
<td>C-Relation</td>
<td>No</td>
<td>Yes</td>
<td>254</td>
<td>String</td>
<td>Left</td>
<td>4.729&quot;</td>
<td>Unformatted</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>colXRel</td>
<td>X-Relation</td>
<td>No</td>
<td>Yes</td>
<td>254</td>
<td>String</td>
<td>Left</td>
<td>4.729&quot;</td>
<td>Unformatted</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>colLength</td>
<td>Length</td>
<td>No</td>
<td>Yes</td>
<td>3</td>
<td>Number</td>
<td>Right</td>
<td>0.7&quot;</td>
<td>Unformatted</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>colScale</td>
<td>Scale</td>
<td>No</td>
<td>Yes</td>
<td>2</td>
<td>Number</td>
<td>Right</td>
<td>0.7&quot;</td>
<td>Unformatted</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>colItemType</td>
<td>Type</td>
<td>No</td>
<td>Yes</td>
<td>2</td>
<td>String</td>
<td>Left</td>
<td>4.729&quot;</td>
<td>Unformatted</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Editable? No
Maximum Data Length: Default
Data Type: Number
Justify: Left
Width: 1.2"
Format: Unformatted
Country: Default
Message Actions

Window Variables
Boolean: bAttributes

Message Actions
On SAM_Create

<table>
<thead>
<tr>
<th>Create Elementary Data Type Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set hWndGloDetailTable = hWndItem</td>
</tr>
<tr>
<td>Set strColType[0] = 'Character Field'</td>
</tr>
<tr>
<td>Set strColType[1] = 'Variable Length Field'</td>
</tr>
<tr>
<td>Set strColType[2] = 'Long Character Field'</td>
</tr>
<tr>
<td>Set strColType[3] = 'Integer'</td>
</tr>
<tr>
<td>Set strColType[4] = 'Decimal'</td>
</tr>
<tr>
<td>Set strColType[5] = 'Number'</td>
</tr>
<tr>
<td>Set strColType[6] = 'Small Integer'</td>
</tr>
<tr>
<td>Set strColType[7] = 'Real'</td>
</tr>
<tr>
<td>Set strColType[8] = 'Floating Point Number'</td>
</tr>
<tr>
<td>Set strColType[9] = 'Double Precision Number'</td>
</tr>
<tr>
<td>Set strColType[10] = 'Date'</td>
</tr>
<tr>
<td>Set strColType[11] = 'Time'</td>
</tr>
<tr>
<td>Set strColType[12] = 'System Time Stamp'</td>
</tr>
</tbody>
</table>

Call SalSetWindowText(hWndForm,'Attributes For '||frmObMgr.strO_ID)

<table>
<thead>
<tr>
<th>Get and save a window handle to the Form's menu bar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set hWndMenuBar = GetMenu(hWndForm)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set the string's length for the GetMenuString() call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call SalStrSetBufferLength(sMenuName, MENU_MAXLENGTH)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Initialize the floating popup menu - use the space bar, while positioned</th>
</tr>
</thead>
<tbody>
<tr>
<td>in the table window &quot;Type&quot; column, to display the popup menu</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>set bFloating = TRUE when Floating popup is displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set bFloating = FALSE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Create a Floating Popup menu for the Form Window</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set hWndFloating = CreatePopupMenu()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>add some menu items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set nGloCount = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>While nGloCount &lt; 13</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>If strColType[nGloCount] = 'Integer' or strColType[nGloCount] = 'Decimal'</th>
</tr>
</thead>
<tbody>
<tr>
<td>add a menu separator</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Call AppendMenu(hWndFloating, MF_Separator, nGloCount, NULL)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Call AppendMenu(hWndFloating, MF_Enabled, nGloCount, strColType[nGloCount])</th>
</tr>
</thead>
</table>

| Set nGloCount = nGloCount+1 |

On SAM_Click

<table>
<thead>
<tr>
<th>If iParam != nAttributeTblRow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call SalTblSetRowFlags(hWndForm,nAttributeTblRow,ROW_Selected,FALSE)</td>
</tr>
<tr>
<td>Set nAttributeTblRow = iParam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>If SalisNull(colFrom)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>If the table row is not inherited, it may be edited—highlight the appropriate column.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Call SalSendMsg(hWndColFocus,MSG_Highlight,0,0)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Call SalEnableWindow(pbDelete)</th>
</tr>
</thead>
</table>

Else

---

- 174 -
Do not allow row to be edited or deleted.

Call SetEnableWindow(pbDelete)

On MSG_Check

† If attributes have been changed, check the table window for any errors with the attributes as they relate to one another.

† If bkDetailsChanged or not bkGloDetailOK
  If bkGloDetailOK
    Call CopyDetails(nGloDetailObject, TRUE)
    If SaveDetails()
      Call SendMessage(hWndDialog, TRUE)
    Else
      Call SendMessage(hWndDialog, FALSE)
  Else
    Call SendMessage(hWndDialog, FALSE)

On MSG_Load

† Fetch all object attributes into the Details table window.

Call GetAliRelatedObjects(nGloDetailObject, nGloRelated, nGloRelatedLevel, nGloRelatedType, strGloObjRelation)

Set bkAttributes = FALSE
Set nGloCount = 0
While nGloRelated(nGloCount) != -1
  If nGloRelatedType(nGloCount) = ENTITY_CrossProduct and nGloRelatedType(nGloCount) = ENTITY_Composite
    RetrieveObjectAttributes(nGloRelated, nGloRelatedLevel, nGloObjRelation)
    Set bkAttributes = TRUE
    Set nGloCount = nGloCount + 1
  Call SendMessage(cmbCompositions, MSG_Load, 0, 0)
  Call SendMessage(cmbCrossProducts, MSG_Load, 0, 0)
If not bkAttributes
  Call SetWindowText(hWndDialog, "Attributes Belonging To " || fmObjMgr.strO_ID || " - (none)"
  Call SendMessage(pbDelete)
  Call SendMessage(pbCompositions)
  Call SendMessage(pbCrossProducts)

† If checking for duplicates, dlgDetails does not show itself. Instead, it traverses the attributes within itself looking for errors. If an error is found, it returns FALSE; otherwise, it returns TRUE.

† If bkGloBuildSQL
  Call CreateIndexes()
  Call CreateTable(nGloDetailObject)
  If bkGloStoreComposites
    Call CreateStoredSQL(ENTITY_Composite)
  If bkGloStoreCrossProducts
    Call CreateStoredSQL(ENTITY_CrossProduct)
  Call SendMessage(hWndDialog, TRUE)
Else
  Call SendMessage(hWndDialog, FALSE)

Else

† If an SQL file is being built, dlgDetails does not show itself. Instead, it traverses the attributes within itself to create indices, tables, and stored SQL statements, and returns TRUE. Otherwise, it shows itself to the user for editing.

† If bkGloDupeCheck
  If NoDuplicateErrors()
    Call SendMessage(hWndDialog, TRUE)
  Else
    Call SendMessage(hWndDialog, FALSE)
Else

† If bkDetailsChanged = TRUE

Background Text: &Composition:
Window Location and Size
On MSG_Load

! Combo box populates itself with all composite entities associated with the object.
! Set nListCount = 0
While nGloRelated[nListCount] != -1
  If nGloRelatedType[nListCount] = ENTITY_Composite
    Call SalNumberToStr(nListCount, 0, strObjLoc)
    Call SalAdd(hWndItem, strGloO_ID[nGloRelated[nListCount]] || SPACES || strObjLoc)
    Set nListCount = nListCount + 1
  If SalQueryCount(hWndItem) = 0
    If nGloObjType[nGloDetailObject] = ENTITY_REGULAR
      Call SalAdd(hWndItem, '(none)')
    Else
      Call SalAdd(hWndItem, '(n/a)')
      CallSalColorSet(hWndItem, COLOR_INDEXWindowText, COLOR_DarkGray)
      Call SalDisableWindow(hWndItem)
      Call SalDisableWindow(hWndCompositions)
      Call SaltoBeSelect(hWndItem, 0)
      Call SalSendMsg(hWndItem, SAM_Click, 0, 0)
On SAM_Click
Call SalQueryText(hWndItem, SalListQuerySelection(hWndItem), strCompObjectID)
Call SalRight(strCompObjectID, 8, strListCount)
Call SalLeft(strCompObjectID, 8, strCompObjectID)
Call SalTrim(strCompObjectID, strCompObjectID)
Call SalTrim(strListCount, strListCount)
Set nListCount = SalToNumber(strListCount)
Pushbutton: pbCompositions
Title: &Define
Window Location and Size
Left: 1.686"
Top: 0.022"
Width: 0.7"
Height: 0.238"
Visible? Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Call SaiSendMsg(cmbCompositions,SAM_Click,0,0)
Set nGloCompNumber = nListCount
If SaIModalDialog(dlgSummaryDetails,hWndForm)
Set bDetailsChanged = TRUE
Background Text: Cross & Product:
Window Location and Size
Left: 2.457"
Top: 0.024"
Width: 1.286"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Combo Box: cmbCrossProducts
Window Location and Size
Left: 2.443"
Top: 0.202"
Width: 1.614"
Height: 1.798"
Visible? Yes
Editable? No
String Type: String
Maximum Data Length: Default
Sorted? Yes
Always Show List? No
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
List Initialization
Message Actions
On MSG_Load
I
! Combo box populates itself with all cross product entries associated with the object.
I
Set nListCount = 0
While nGlo Related[nListCount] /= -1
If nGloRelatedType[nListCount] = ENTITY_CrossProduct
Call SaiNumberToStr(nListCount,0,strObjLoc)
Call SaiListAdd(hWndItem,strGlo number[nListCount]]||SPACES||strObjLoc)
Set nListCount = nListCount + 1
If SaiListQueryCount(hWndItem) = 0
If nGloObjType[nGloDetailObject] = ENTITY_Regular
Call SaiListAdd(hWndItem,'(none)')
Else
Call SaiListAdd(hWndItem,'(val)')
Call SaiColorSet(hWndItem,COLO lndexWindowText,COLO_DarkGray)
Call SaiDisableWindow(hWndItem)
Call SaiListSelSelect(hWndItem,0)
Call SaiSendMsg(hWndItem,SAM_Click,0,0)
On SAM_Click
Call SaiListQueryText(hWndItem,SaiListQuerySelection(hWndItem),strCompObjectId)
Call SaiStrRight(strCompObjectId,8,strListCount)
Call SaiStrLeft(strCompObjectId,8,strCompObjectId)
Call SaiStrTrim(strCompObjectId,strCompObjectId)
Call SaiStrTrim(strListCount,strListCount)
Set nListCount = SaiStrToNumber(strListCount)
Pushbutton: pbCrossProducts
Title: Define
Window Location and Size
Left: 4.086"
Top: 0.202"
Width: 0.7"
Height: 0.238"
Visible? Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Call SendMsg(cmbCrossProducts,SAM_Click,0,0)
Set nGlOCompNumber = nListCount
If SalModelDialog(dlgSummaryDetails,hWndForm)
  Set bDetailsChanged = TRUE
Pushbutton: pbAdd
Title: &Add
Window Location and Size
Left: 4.857"
Top: 0.143"
Width: 0.9"
Height: 0.298"
Visible? Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
  If nAttributeTblRow = SetTblInsertRow(tblDetails, TBL_MaxRow)
    Call SetTblSetFocusRow(tblDetails, nAttributeTblRow)
    Call SetTblSetContext(tblDetails, nAttributeTblRow)
    Call SetTblSetFocusCell(tblDetails, nAttributeTblRow, colName, -1, -1)
    Call SetWindowText(hWndForm,"Attributes Belonging To ",
istrObjMgr.strO_ID)
    Call EnableWindow(pbDelete)
    If cmbCompositions = (none)
      Call EnableWindow(pbCompositions)
    If cmbCrossProducts = (none)
      Call EnableWindow(pbCrossProducts)
    Set bDetailsChanged = TRUE
Pushbutton: pbDelete
Title: &Remove
Window Location and Size
Left: 5.6"
Top: 0.143"
Width: 0.9"
Height: 0.298"
Visible? Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
  If nAttributeTblRow = SetTblInsertRow(tblDetails)
    Call SetTblSetFocusRow(tblDetails, nAttributeTblRow)
    Call SetTblSetContext(tblDetails, nAttributeTblRow)
    Set colName = (unnamed)
    If SalMessageBox("Are you sure you want to remove the ",
    strColName, attribute from ",
    strObjMgr.strO_ID) = YES
      Set colName = (unnamed)
      Call SetTblSelectRow(tblDetails, nAttributeTblRow, TBL_NoAdjust)
      Set bDetailsChanged = TRUE
      Call EnableWindow(pbDelete)
If colName = NULL
    Set nAttributeTblRow = nAttributeTblRow - 1
    Call SalTblSetContext(tblDetails,nAttributeTblRow)
    Call SalTblSetFocusRow(tblDetails,nAttributeTblRow)
If colFrom = NULL
    Call SalDisableWindow(pDelete)
If not SalTblAnyRows(tblDetails,O,O)
    Call SalSetWindowText(hWndDialog,'Attributes Belonging To "[frmObjMgr.strO_ID]" - (none)')
    Call SalDisableWindow(pDelete)
    Call SalDisableWindow(pCompositions)
    Call SalDisableWindow(pCrossProducts)
Else
    If colName = (unnamed)
        Call SalClearField(colName)
    Call SalTblSetFocusCell(tblDetails,nAttributeTblRow,colName,-1,-1)
If colName = NULL
    Set nAttributeTblRow = nAttributeTblRow - 1
Pushbutton: pbOk
    Title: OK
Window Location and Size
    Left: 6.771"
    Top: 0.143"
    Width: 0.9" 
    Height: 0.298"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
    Call SalSendMsg(tblDetails,MSG_Check,0,0)
Pushbutton: pbCancel
    Title: Cancel
Window Location and Size
    Left: 7.714"
    Top: 0.143"
    Width: 0.9"
    Height: 0.296"
Visible? Yes
Keyboard Accelerator: Esc
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
    If not bSqlDetailsOK
        Call RestoreDetails(nGloCell[nGloAbsPos])
    Set bSqlDetailsOK = TRUE
    Call SalEndDialog(hWndDialog,FALSE)
On SAM_KillFocus
    Call SalPostMsg(tblDetails,colName,MSG_Highlight,0,0)
Window Variables
    Boolean: bSpecificsDialog
    Number: nAttributeTblRow
    Number: nCheck
    Number: nLength
    Number: nListCount
    Number: nScale
    String: strCheck
    String: strColType[15]
    String: strCompObjectID
    String: strListCount
    String: strObjectLoc
Window Handle: hWndColFocus
Window Handle: hWndDialog
1 Floating Menu Variables
    Number: nMenuID
    String: strMenuName
Window Handle: hWndMenuBar
Window Handle: hWndSubMenu
Window Handle: hWndSystemMenu
Window Handle: hWndSystemSubMenu

I
for the floating popup menu
Boolean: bFloating
Boolean: bDetailsChanged
Window Handle: hWndFloating

Message Actions
On SAM_Create
  Call SaveGrabCursor(TRUE)
  Call SaveHideWindow(hWndForm)
  Set bDetailsChanged = FALSE
  Set hWndDialog = hWndForm
  Call SaveSetWindowText(hWndForm,'Attributes Belonging To \[strGloO_ID\[nGloDetailObject])
  Call SavePostMsg(tbIDetalls,MSG_Load,0,0)
On WM_COMMAND
  If bFloating
    If wParam > -1 and wParam < 13
      Call strReplace(strColType[wParam],strColType[wParam],'&'), 1, NULL, tblDetalls.colItem)
      Call strMid(TYPE_CHOICES,wParam,1,strCheck)
      Set tblDetails.colItemType = wParam
    Set bFloating = FALSE
  On MSG_Show
    Set bFloating = TRUE
  Call ClientToScreen(hWndForm, lParam)
  Call TrackPopupMenu(hWndFloating, 0, SaveNumberLow(lParam), SaveNumberHigh(lParam), hWndForm, NULL)
  If WM_COMMAND's wParam Is the menu item chosen when bFloating = TRUE
On SAM_Close
  Call SaveSendMsg(pbcancel,SAM_Click,0,0)

Dialog Box: digDisassociate
Title: Delete An Association
Display Settings
  Visible at Design time? No
  Type of Dialog: Modal
  Window Location and Size
    Left: 0.625"
    Top: 0.625"
    Width: 5.086"
    Height: 2.857"
  Absolute Screen Location? Yes
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Default

Contents
  Background Text: Associated &Object:
  Window Location and Size
    Left: 0.167"
    Top: 0.202"
    Width: 1.629"
    Height: 0.167"
    Visible? Yes
    Justify: Left
    Font Name: Default
    Font Size: Default
    Font Enhancement: Default
    Text Color: Default
    Background Color: Default
  ComboBox: cbObject
  Window Location and Size
    Left: 1.938"
    Top: 0.156"
    Width: 1.4"
    Height: 1.927"
On SAM_Create

Set strGloCSV = strGloObjAssociations[nGloCell[nGloAbsPos]]
Set nListCount = 0
While SalStrLength(strGloCSV) > 1
    Set strObjectID = ReadCSV()
    Call SalStrLeft(strObjectID,SalStrScan(strObjectID,',',strObjLoc)
    Call SalStrTrim(strObjLoc)
    Set nObjLoc = SalStrToNumber(strObjLoc)
    Set strLocList[nListCount] = strObjLoc
    Call SalStrRight(strObjectID,1,strObjLoc[nListCount])
    If strObjLoc[nListCount] > 'Z'
        Call SalNumberToStr(nListCount,0,strListCount)
        Set strObjectID = strGlo_ID[nObjLoc]
        Call SalListAdd(cbObject,strObjectID[SPACES][strListCount])
    Set nListCount = nListCount + 1
    Call SalPostMsg(hWndForm,MSG_Created,O,O)
On SAM_Click

If strAssocType = 'g'
    Set strAssociationType = 'Generalization'
    Set mAssociation = 'Every instance of a(n) "'strObjectID" is a kind of "'strObjectID". All attributes belonging to the "'strObjectID" are inherited by the "'strObjectID".'
    If strAssocType = 'a'
        Set strAssociationType = 'Aggregation'
        Set mAssociation = 'Every instance of a(n) "'strObjectID" can be associated with a predefined "'strObjectID". Key attributes used to identify the "'strObjectID" are retained by the "'strObjectID".'
        If strAssocType = 'o'
            Set strAssociationType = 'Interaction (one to one)'
            Set mAssociation = 'Every "'strObjectID" can have only one "'strObjectID". Every "'strObjectID" can have only one "'strObjectID". Key attributes used to identify the "'strObjectID" are retained by the "'strObjectID".'
            If strAssocType = 'm'
                Set strAssociationType = 'Interaction (many to one)'
                Set mAssociation = 'Every "'strObjectID" can have one or more "'strObjectID"s. Every "'strObjectID" can have only one "'strObjectID". Key attributes used to identify the "'strObjectID" are retained by the "'strObjectID".'
                If strAssocType = 'c'
                    Set strAssociationType = 'Composition'
                    Set mAssociation = 'The "'strObjectID" is partially defined by the set of all "'strObjectID"s. Selected attributes of all "'strObjectID"s are summarized into "'strObjectID".'
                    If strAssocType = 'x'
                        Set strAssociationType = 'x'}
Set strAssociationType = 'Cross Product'
Set mlAssociation = The listObjectID is partially defined by the listObject[ID]. Key attributes used to identify the listObject[ID] are retained as part of the listObject[ID]. Numeric attributes belonging to listObject[ID] can be summarized by listObject[ID].

Background Text: Association Type:
Window Location and Size
Left: 0.186”
Top: 0.917”
Width: 1.529”
Height: 0.167”
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strAssociationType
Data
Maximum Data Length: Default
Data Type: String
Editable? No
Display Settings
Window Location and Size
Left: 1.757”
Top: 0.929”
Width: 3.029”
Height: 0.25”
Visible? Yes
Border? No
Justify: Left
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Multiline Field: mlAssociation
Data
Maximum Data Length: Default
String Type: String
Editable? No
Display Settings
Border? No
Word Wrap? Yes
Vertical Scroll? No
Window Location and Size
Left: 0.3”
Top: 1.536”
Width: 4.343”
Height: 0.762”
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Pushbutton: pbOk
Title: Ok
Window Location and Size
Left: 3.588”
Top: 0.156”
Width: 1.2”
Height: 0.292”
Visible? Yes
Keyboard Accelerator: Enter
1 Association is deleted from the strGloObjectAssociations[*] array, and the workspace is refreshed.

1
Call SalWaitCursor(TRUE)
Set strGloCSV = strGloObjectAssociations[nGloCell[nGloAbsPos]]
Call SalStrReplace(strGloCSV,SalStrScan(strGloCSV,'#')strGloList[nListCount]), SalStrLength(strGloList[nListCount]) + 4,NULL,strGloCSV
Set strGloObjectAssociations[nGloCell[nGloAbsPos]] = strGloCSV
Set strGloCSV = strGloObjectAssociations[nObjLoc]
Call SalNumberToSlr(nGloCell[nGloAbsPos],0,strObjLoc)
Call SalStrReplace(strGloCSV,SalStrScan(strGloCSV,'#')strObjLoc), SalStrLength(strObjLoc) + 4,NULL,strGloCSV
Set strGloObjectAssociations[nObjLoc] = strGloCSV
Set nGloCount = 1
While nGloCount < 11
If nGloCell(nGloScreen*10 + nGloCount) = nObjLoc
Set nRelPos = SalNumberMod(nGloAbsPos,10)
If nRelPos = 0
Set nRelPos = 10
If nRelPos < nGloCount
Call SalSendMsg(salWndForm(hWndObjMgr.nWndDF[nRelPos],MSG_HideLine,nGloCount,0)
Else
Call SalSendMsg(salWndForm(hWndObjMgr.nWndDF[nGloCount],MSG_HideLine,nRelPos,0)
Set nGloCount = 11
Else
Set nGloCount = nGloCount + 1
Call RefreshLabels()
Call SalWaitCursor(FALSE)
Set bGloChanged = TRUE
Call SalEndDialog(hWndForm,TRUE)

Pushbutton: pbCancel
Title: Cancel
Window Location and Size
Left: 3.586"
Top: 0.5"
Width: 1.2"
Height: 0.298"
Visible? Yes
Keyboard Accelerator: Esc
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Call SalEndDialog(hWndForm,FALSE)

Group Box: Description
Window Location and Size
Left: 0.129"
Top: 1.266"
Width: 4.686"
Height: 1.131"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Window Variables
Number: nListCount
String: strListCount
Number: nObjLoc
String: strObjLoc
String: strLocList[*]
String: strObjRel[*]
String: strObjectID
String: strAssocType
Number: nRelPos

Message Actions
On SAM Create
  Call SetWaitCursor(TRUE)
  Call SetHideWindow(hWndForm)
  Call SetWindowText(hWndForm,'Disassociate An Object From ' || frmObjMgr.strO_ID)
On MSG_CREATED
  Call SetWaitCursor(FALSE)
  If there exist no objects to associate with the object having focus, this window displays a message,
  and does not show.
  If not SetListSetSelect(cbObject,0)
    Call SetMessageBox('There are no high level objects dependent upon ' || frmObjMgr.strO_ID || ' to
disassociate.','APPNAME,
    MB_OK|MB_ICONASTERISK)
    Call SetEndDialog(hWndForm,FALSE)
Else
  Call SetSendMsg(cbObject,SAM_Click,0,0)
  Call SetShowWindow(hWndForm)
On SAM_CLOSE
  Call SetSendMsg(pbCancel,SAM_Click,0,0)

Dialog Box: dlgFindObject
Title: OSAM* Object List
Display Settings
  Visible at Design time? No
  Type of Dialog: Modal
Window Location and Size
  Left: 1.125"
  Top: 0.75"
  Width: 3.657"
  Height: 2.714"
Absolute Screen Location? Yes

Contents
  Background Text: &Show:
    Window Location and Size
    Left: 0.1"
    Top: 0.071"
    Width: 0.614"
    Height: 0.167"
  Visible? Yes
  Justify: Left
  Font Name: Default
  Font Size: Default
  Font Enhancement: Default
  Text Color: Default
  Background Color: Default

Combo Box: strObjType
Window Location and Size
  Left: 0.786"
  Top: 0.06"
  Width: 2.586"
  Height: 1.488"
  Visible? Yes
  Editable? No
  String Type: String
  Maximum Data Length: Default
  Sorted? No
  Always Show List? No
  Vertical Scroll? Yes
  Font Name: Default
  Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
List Initialization
Text: Every Object
Text: Regular Entities
Text: Simple Domains
Text: Cross Product Entities
Text: Composite Entities
Message Actions
On SAM_Create
Call ListSetSelect(hWndItem,ALL)
Call ListInsert(hWndItem,1,'Dependent Upon '][frmObjMgr.strO_ID)
Call ListInsert(hWndItem,-1,'Depended on by '['][frmObjMgr.strO_ID)
Call SendMsg(lbObjects,MSG_Load,O,O)
If nObjType = ALL
Call ShowWindow(hWndForm)
On SAM_Click
Call PostMsg(lbObjects,MSG_Load,O,O)
Background Text: &Object:
Window Location and Size
Left: 0.1"
Top: 0.357"
Width: 0.643"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Data Field: strTargetID
Data
Maximum Data Length: Default
Data Type: String
Editable? No
Display Settings
Window Location and Size
Left: 0.796"
Top: 0.321"
Width: 1.471"
Height: 0.25"
Visible? Yes
Border? Yes
Justify: Left
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
List Box: lbObjects
Window Location and Size
Left: 0.9"
Top: 0.56"
Width: 1.357"
Height: 1.726"
Visible? Yes
Multiple selection? No
Sorted? Yes
Vertical Scroll? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

- 185 -
List Initialization
Message Actions

On MSG_Load

I Lists all objects of a certain type in the combo box.
I
  Call SaILWaitCursor(TRUE)
  Set nObjType = SaILListQuerySelectlon(strObjType)
  Call SaILListClear(hWndItem)
  Call SaILListRedraw(hWndItem, FALSE)
If nObjType < DEPENDENT_UPON
  Set nGloCount = 1
  While nGloObjType[nGloCount] != 0
      If nGloObjCell[nGloCount] = nObjType
          Call SaILNumberToStr(nGloCount, 0, strObject)
          Set strObject = strGloObj_ID[nGloCount]||TAB||strObject
          Call SaILListAdd(hWndItem, strObject)
          Call SaILListRedraw(hWndItem, TRUE)
      Else
          If nObjType = ALL or nGloObjType[nGloCount] = nObjType
              Call GetAIIRelatedObjects(nGloCell(nGloAbsPos), FALSE, nGloRelated, nGloRelatedLevel, nGloRelatedType, strGloObjRelation)
          Else
              Call GetAIIRelatedObjects(nGloCell(nGloAbsPos), TRUE, nGloRelated, nGloRelatedLevel, nGloRelatedType, strGloObjRelation)
          Call SaILListClear(hWndItem)
          Set nGloCount = 1
          While nGloRelated[nGloCount] != -1
              Call SaILNumberToStr(nGloRelated[nGloCount], 0, strObject)
              Call SaILListAdd(hWndItem, strObject)
          Call SaILWaitCursor(FALSE)
          Call SaILListRedraw(hWndItem, TRUE)
Call SaILvalidateWindow(hWndItem)
If SaILListQueryCount(hWndItem) = 0
If nObjType = ALL
    Call SaILMessageBox('No Objects Have Been Created!', APPNAME, MB_OK|MB_ICONASTERISK)
    Set nObjType = -1
    Call SaISendMsg(pbCancel, SAM_Click, 0, 0)
    Return FALSE
    Set strTargetID = '(none)'
Else
    Call SaILSetSelected(hWndItem, 0)
    Call SaISendMsg(hWndItem, SAM_Click, 0, 0)
On SAM_DoubleClick
    Call SaISendMsg(pbOk, SAM_Click, 0, 0)
On SAM_Click
    Call SaILListQueryText(lbObjects, SaILListQuerySelectlon(lbObjects), strObject)
    Call SaIStrTokenize(strObject, TAB, TAB, strObjParm)
    Set strTargetID = strObjParm[0]
    Call SaILWaitCursor(FALSE)
Pushbutton: pbFind
Title: &Flnd
Window Location and Size
Left: 2.543"
Top: 0.714"
Width: 0.514"
Height: 0.238"
Visible? No
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
    Call SaISendMsg(lbObjects, MSG_Load, 0, 0)
Pushbutton: pbOk
Title: Ok
Window Location and Size
Left: 2.414"
Top: 1.69"
Width: 0.986"
Height: 0.274"
Visible: Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
I
I Verffies that an object has been selected for finding.
I
If SailsNull(strTargetID)
   Call SalMessageBox('Please Select An Object From The Object List.',APPNAME,MB_Ok,MB_IconAsterisk)
   Call SalSetFocus(lbObjects)
   Return FALSE
If strTargetID = '(none)'
   Call SalMessageBox('There are no other objects which are ']strObjType[[',APPNAME,
       MB_Ok,MB_IconAsterisk)
   Call SalSetFocus(strObjType)
   Return FALSE
I
I Finds object in workspace, and repaints workspace to show object.
I
Set nGloCount = SalStrToNumber(strObjParm[1])
Set nHbar = nGloObjHBar[nGloCount]
Set nVbar = nGloObjVBar[nGloCount]
Set df nHbar = nHbar-MAX_SCROLL/2
Set df nVbar = nVbar-MAX_SCROLL/2
Call SalScrollSetPos(hbar,nHbar)
Call SalScrollSetPos(vBar,nVbar)
Set nGloAbsPos = nGloObjCell[nGloCount]
Set nGloScreen = (nVbar*(MAX_SCROLL+1)+nHbar
Call SalPostMsg(frmObjMgr.pbOrgln,MSG_Redraw,0,0)
Call SalEndDialog(hWndForm,TRUE)

Pushbutton: pbCancel
Title: Cancel
Window Location and Size
Left: 2.414"
Top: 2.012"
Width: 1.0"
Height: 0.274"
Visible: Yes
Keyboard Accelerator: (none)
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click
Call SalEndDialog(hWndForm,FALSE)

Window Variables
Number: nObjType
String: strDirection
String: strObject
String: strObjParm[*]

Message Actions
On SAM_Create
Call SalSetWaitCursor(TRUE)
Call SalHideWindow(hWndForm)
On SAM_Destroy
Call SalSetWaitCursor(FALSE)
On SAM_Close
Call SalSendMsg(pbCancel,SAM_Click,0,0)
Dialog Box: dlgRules

Title:

Display Settings
Visible at Design time? No
Type of Dialog: Modal
Window Location and Size
Left: 0.638"
Top: 0.625"
Width: 5.129"
Height: 4.095"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Contents
Background Text: On &Insert:
Window Location and Size
Left: 0.186"
Top: 0.083"
Width: 0.886"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Multiline Field: mllnsert
Data
Maximum Data Length: 254
String Type: String
Editable? Yes

Display Settings
Border? Yes
Word Wrap? Yes
Vertical Scroll? Yes
Window Location and Size
Left: 0.186"
Top: 0.286"
Width: 4.657"
Height: 0.81"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Message Actions
On SAM_Create
Set mllnsert = strGloObj[nGloCell[nGloAbsPos]]
On SAM.Validate
Call ReplaceChar(mllnsert,DEL,SPACE)
Call ReplaceChar(mllnsert,",",""
Return VALIDATE_OK

Background Text: On &Update:
Window Location and Size
Left: 0.186"
Top: 1.167"
Width: 1.029"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default

- 188 -
Background Color: Default
Multiline Field: mlUpdate
Data
Maximum Data Length: 254
String Type: String
Editable? Yes
Display Settings
Border? Yes
Word Wrap? Yes
Vertical Scroll? Yes
Window Location and Size
Left: 0.186"
Top: 1.309"
Width: 4.657"
Height: 0.81"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
Set mlUpdate = strGloObjUpdate[nGloCell[nGloAbsPos]]
On SAM_Validate
Call ReplaceChar(mlUpdate,DEL,SPACE)
Call ReplaceChar(mlUpdate,"","")
Return VALIDATE_OK

Background Text: On &Delete:
Window Location and Size
Left: 0.186"
Top: 2.25"
Width: 0.886"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Multiline Field: mlDelete
Data
Maximum Data Length: 254
String Type: String
Editable? Yes
Display Settings
Border? Yes
Word Wrap? Yes
Vertical Scroll? Yes
Window Location and Size
Left: 0.186"
Top: 2.452"
Width: 4.657"
Height: 0.81"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On SAM_Create
Set mlDelete = strGloObjDelete[nGloCell[nGloAbsPos]]
On SAM_Validate
Call ReplaceChar(mlDelete,DEL,SPACE)
Call ReplaceChar(mlDelete,"","")
Return VALIDATE_OK

Pushbutton: pbOk

- 189 -
Title: OK
Window Location and Size
  Left: 2.371"
  Top: 3.345"
  Width: 1.2"
  Height: 0.298"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
  On SAM_Click
    Set strGloObjInsert[nGloCell[nGloAbsPos]= mInsert
    Set strGloObjUpdate[nGloCell[nGloAbsPos] = mUpdate
    Set strGloObjDelete[nGloCell[nGloAbsPos] = mDelete
    Call SaiEndDialog(hWndForm,TRUE)

Pushbutton: pbCancel
Title: Cancel
Window Location and Size
  Left: 3.643"
  Top: 3.345"
  Width: 1.2"
  Height: 0.298"
Visible? Yes
Keyboard Accelerator: Esc
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
  On SAM_Click
    Call SaiEndDialog(hWndForm, FALSE)

Window Variables
Message Actions
  On SAM_Create
    ! Accepts Insert, update, and delete rules for a given object.
    !
    Call SaiSetWindowText(hWndForm,"Rules For "[frmObjMgr.strO_ID"
  On SAM_Close
    Call SaiSendMessage(pbCancel,SAM_Click,0,0)

Dialog Box: dlgSummaryDetails
Title:
Display Settings
  Visible at Design time? No
  Type of Dialog: Modal
Window Location and Size
  Left: 0.85"
  Top: 0.856"
  Width: 5.557"
  Height: 4.31"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Contents
Check Box: cbCounts
Title:
Window Location and Size
  Left: 0.057"
  Top: 0.048"
  Width: 5.071"
  Height: 0.25"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
On &SAM_Create
If nGloRelatedType[nGloCompNumber] = ENTITY_Composite
  Call SetSetWindowText(hWndItem, "&Show " & frmObjMgr.strO_ID & " count on
'" & strGloO_ID[iGloRelated[iGloCompNumber]] & ")"
Else
  Call SetSetWindowText(hWndItem, "&Show " & frmObjMgr.strO_ID & " counts broken out by
'" & strGloO_ID[iGloRelated[iGloCompNumber]] & ")"
Table Window: tblSummaryDetails
Title: 
Icon File: 
Display Settings
Visible at Design time? Yes
Automatically Created at Runtime? Yes
Initial State: Normal
Maximizable? No
Minimizable? No
System Menu? No
Resizable? No
Window Location and Size
Left: 0.0"
Top: 0.333"
Width: 5.414"
Height: 3.226"
Visible? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Memory Settings
Maximum Rows In Memory: Default
Discardable? Yes
Menu
Contents
Column: colCompValue
Title: Numeric Attribute
Visible? Yes
Editable? No
Maximum Data Length: 18
Data Type: String
Justify: Left
Width: 1.729"
Format: Unformatted
Country: Default
Message Actions
On MSG_Highlight
Call SalTblSetFocusCell(hWndForm, nAttrSummaryTblRow, hWndItem, -1, -1)
Column: colCompItemType
Title: Type
Visible? No
Editable? No
Maximum Data Length: 50
Data Type: Number
Justify: Left
Width: 2.857"
Format: Unformatted
Country: Default
Message Actions
Column: colCompTblRow
Title:
Visible? No
Editable? Yes
Maximum Data Length: Default
Data Type: Number
Justify: Left
Width: 1.2"
Format: Unformatted
Country: Default
Message Actions
Column: colLowest
Title: Lowest
Visible? Yes
Editable? Yes
Maximum Data Length: Default
Data Type: String
Justify: Center
Width: 0.786
Format: Unformatted
Country: Default
Message Actions
On SAM_Create
Set hWndSummaryFocus = hWndItem
On SAM_AnyEdit
If not SalisNull(hWndItem)
    If colLowest = "Y" or colLowest = 'Y'
        Set colLowest = 'Yes'
        Return TRUE
    If colLowest = "N" or colLowest = 'n'
        Set colLowest = 'No'
        Return TRUE
    Call SalIClearField(hWndItem)
On SAM_KillFocus
If SalisNull(hWndItem)
    Call SalSetWindowText(hWndItem,'No')
On MSG_Highlight
Call SalTblSetFocusCell(hWndForm,nAttrSummaryTblRow,hWndItem,-1,-1)
On SAM_SetFocus
Set hWndSummaryFocus = hWndItem
Column: colHighest
Title: Highest
Visible? Yes
Editable? Yes
Maximum Data Length: Default
Data Type: String
Justify: Center
Width: 0.786
Format: Unformatted
Country: Default
Message Actions
On SAM_AnyEdit
If not SalisNull(hWndItem)
    If colHighest = "Y" or colHighest = 'Y'
        Set colHighest = 'Yes'
        Return TRUE
    If colHighest = "N" or colHighest = 'n'
        Set colHighest = 'No'
        Return TRUE
    Call SalIClearField(hWndItem)
On MSG_Highlight
Call SalTblSetFocusCell(hWndForm,nAttrSummaryTblRow,hWndItem,-1,-1)
On SAM_KillFocus
If SalisNull(hWndItem)
    Call SalSetWindowText(hWndItem,'No')
On SAM_SetFocus
Set hWndSummaryFocus = hWndItem
Column: colAverage
Title: Average
Visible? Yes
Editable? Yes
Maximum Data Length: Default
Data Type: String
Justify: Center
Width: 0.786
Format: Unformatted
Country: Default
Message Actions
On SAM_AnyEdit
  If not SalIsNul(hWndItem)
    If colAverage = 'Y' or colAverage = 'Y'
      Set colAverage = 'Yes'
      Return TRUE
    If colAverage = 'N' or colAverage = 'n'
      Set colAverage = 'No'
      Return TRUE
  Call SalIClearField(hWndItem)
On MSG_Highlight
  Call SalITblSetFocusCell(hWndForm,nAttrSummaryTblRow,hWndItem,-1,-1)
On SAM_KillFocus
  If SalIsNul(hWndItem)
    Call SalSetWindowText(hWndItem,'No')
On SAM_SetFocus
  Set hWndSummaryFocus = hWndItem
Column: colTotal
  Title: Total
  Visible? Yes
  Editable? Yes
  Maximum Data Length: Default
  Data Type: String
  Justify: Center
  Width: 0.766"
  Format: Unformatted
  Country: Default
Message Actions
On SAM_AnyEdit
  If not SalIsNul(hWndItem)
    If colTotal = 'Y' or colTotal = 'Y'
      Set colTotal = 'Yes'
      Return TRUE
    If colTotal = 'N' or colTotal = 'n'
      Set colTotal = 'No'
      Return TRUE
  Call SalIClearField(hWndItem)
On MSG_Highlight
  Call SalITblSetFocusCell(hWndForm,nAttrSummaryTblRow,hWndItem,-1,-1)
On SAM_KillFocus
  If SalIsNul(hWndItem)
    Call SalSetWindowText(hWndItem,'No')
On SAM_SetFocus
  Set hWndSummaryFocus = hWndItem
Window Variables
Message Actions
On SAM_Click
  If IParam 1 = nAttrSummaryTblRow
    Call SalITblSetRowFlags(hWndForm,nAttrSummaryTblRow,ROW_Selected,FALSE)
    Set nAttrSummaryTblRow = IParam
  Call SAI SendMsg(hWndSummaryFocus,MSG_Highlight,0,0)
On SAM_EndCellTab
  Call SalPostMsg(coiLowest,MSG_Highlight,0,0)
On MSG_Load
  // Populates the summary attributes table with all numeric attributes, and their summary settings
  // (from colCRel or colXRel in the digDetails attribute table.)
  //
  Call SalNumberToStr(nGloRelated{nGloCompNumber},strCompNumber)
  If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
    Set strCompDef = strGloObjXRel{nGloDetailObject}
  Else
    Set strCompDef = strGloObjCRel{nGloDetailObject}
  If SAI StrScan(strCompDef,strCompNumber) = -1
    Set cbCounts = TRUE
  Else
    Set cbCounts = FALSE
    Set bCompAttributes = FALSE
    Set nDetailTblRow = TBL_MinRow
While SaITbIFindNextRow(hWndGloDetailTable, nDetailTblRow, 0, 0)
Call SaITbISetContext(hWndGloDetailTable, nDetailTblRow)
If collItem_Type > 2 and collItem_Type < 10 and collForm = NULL
    Set nAttrSummaryTblRow = SaITblInsertRow(hWndForm, TBL_MaxRow)
Call SaITbISetContext(hWndForm, nAttrSummaryTblRow)
Call SaITbISetRowFlags(hWndForm, nAttrSummaryTblRow, ROW_New, FALSE)
Set colCompTblRow = nDetailTblRow
Set colCompValue = hWndGloDetailTable.colName
Set colCompItem_Type = hWndGloDetailTable.colItem_Type
If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
    Set strCompDef = hWndGloDetailTable.colXRRel
Else
    Set strCompDef = hWndGloDetailTable.colCRel
If SaIStrScan(strCompDef, '#', nStrCompNumber) != -1
    Call SalStrMid(strCompDef, SalStrScan(strCompDef, '#', nStrCompNumber), SalStrLength(strCompNumber) + 7, strCompDef)
Else
    Set strCompDef = NULL
If SalStrScan(strCompDef, 'L', nStrCompNumber) != -1
    Set colLowest = 'Yes'
Else
    Set colLowest = 'No'
If SalStrScan(strCompDef, 'H', nStrCompNumber) != -1
    Set colHighest = 'Yes'
Else
    Set colHighest = 'No'
If SalStrScan(strCompDef, 'A', nStrCompNumber) != -1
    Set colAverage = 'Yes'
Else
    Set colAverage = 'No'
If SalStrScan(strCompDef, 'T', nStrCompNumber) != -1
    Set colTotal = 'Yes'
Else
    Set colTotal = 'No'
Set bCompAttributes = TRUE
Call SalWaitCursor(FALSE)
If bCompAttributes
    Call SaISendMsg(hWndForm, SAM_C1ick, 0, 0)
Else
    Call SalMessageBoxfNo numeric attributes have been defined for '{frmObjMgr.strO_ID}.
APPNAME, MB_OK
ColonAsterisk)
On MSG_Check
    Updates the dlgDetails attribute summary column (colCRel or colXRRel) to reflect changes made
to summaries.
    Set nAttrSummaryTblRow = TBL_MinRow
While SaITbIFindNextRow(hWndForm, nAttrSummaryTblRow, 0, 0)
Call SaITbISetContext(hWndForm, nAttrSummaryTblRow)
Call SaITbISetContext(hWndGloDetailTable, colCompTblRow)
If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
    Set strCompDef = hWndGloDetailTable.colXRRel
Else
    Set strCompDef = hWndGloDetailTable.colCRel
If SalStrScan(strCompDef, '#', nStrCompNumber) != -1
    Call SalStrReplace(strCompDef, '#', nStrCompNumber),
    SalStrScan(strCompDef, '#', nStrCompNumber),
    SalStrLength(strCompNumber) + 7, NULL, strCompDef)
    Call SalStrRight(strCompDef, 1, strComma)
    If strComma = ','
        Set strCompDef = strCompDef;'
    Set strCompDef = strCompDef[#strCompNumber#];'
    If colLowest = 'Yes'
        Set strCompDef = strCompDef}'L'
    Else
        Set strCompDef = strCompDef}'H'
    If colHighest = 'Yes'
        Set strCompDef = strCompDef}'H'
    Else
        Set strCompDef = strCompDef}'H'
    If strCompDef = strCompDef}'H'}
Set strCompDef = strCompDef[;]'
If colAverage = 'Yes'
    Set strCompDef = strCompDef[']A'
Else
    Set strCompDef = strCompDef[;]'
If colTotal = 'Yes'
    Set strCompDef = strCompDef[']T'
Else
    Set strCompDef = strCompDef[;]'
Set strCompDef = strCompDef[;]'
If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
    Set hWndGloDetailTable.colXRel = strCompDef
Else
    Set hWndGloDetailTable.colXRel = strCompDef[;]'
If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
    Set strCompDef = strGloObjXRel[nGloDetailObject]
Else
    Set strCompDef = strGloObjXRel[nGloDetailObject]
If SaIStrScan(strCompDef,strCompNumber) = -1
    Call SaIStrReplace(strCompDef,strCompNumber,
                         SaIStrScan(strCompDef,strCompNumber),
                         SaIStrLength(strCompNumber)+1,NULL,strCompDef)
If cbCounts
    If strCompDef = NULL
        Set strCompDef = strCompDef[;]'
    Set strCompDef = strCompDef[;]'
    If nGloRelatedType[nGloCompNumber] = ENTITY_CrossProduct
        Set strGloObjXRel[nGloDetailObject] = strCompDef
    Else
        Set strGloObjXRel[nGloDetailObject] = strCompDef
Pushbutton: pbOk
Title: Ok
Window Location and Size
    Left: 2.657"
    Top: 3.619"
    Width: 1.229"
    Height: 0.298"
Visible: Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
    On SAM_Click
        Call SISendMsg(tbISummaryDetails,MSG_Check,0,0)
        Call SaIEndDialog(hWndForm,TRUE)
Pushbutton: pbCancel
Title: Cancel
Window Location and Size
    Left: 3.957"
    Top: 3.619"
    Width: 1.229"
    Height: 0.298"
Visible: Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
    On SAM_Click
        Call SaIEndDialog(hWndForm,FALSE)
Window Variables
    Boolean: bCompAttributes
    Number: nAttrSummaryTblRow
    Number: nDetailTblRow
    Number: nListCount
    Number: nSettings
    String: strComma
    String: strCompDef
String: strCompNumber
String: strCompObjectID
String: strCountDef
String: strObject
String: strObjList[5]
String: strObjLoc

Window Handle: hWndSummaryFocus

Message Actions
On SAM_Create
Call SalWaitCursor(TRUE)

If nGloRelatedType[nGloCompNumber] = ENTITY_Composite
Call SalSetWindowText(hWndForm,'Composition [[strGloO_ID[nGloRelated[nGloCompNumber]]]] Defined For
[[frmObjMgr.strO_ID])
Else
Call SalSetWindowText(hWndForm,'Cross Product YYYY [[strGloO_ID[nGloRelated[nGloCompNumber]]]] Defined For
[[frmObjMgr.strO_ID])

Call SalPostMsg(tblSummaryDetails,MSG_Load,0,0)
Call SalPostMsg(tblSummaryDetails,MSG_Created,0,0)

On SAM_Close
Call SalSendMsg(pbCancel,SAM_Click,0,0)

Dialog Box: dlgTypeSpecifics
Title: Type Specifics
Display Settings
Visible at Design time? No
Type of Dialog: Modal
Window Location and Size
Left: 0.625"
Top: 0.625"
Width: 6.8"
Height: 1.25"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Contents
Data Field: dfDescription1

Data

Maximum Data Length: Default
Data Type: String
Editable? No

Display Settings
Window Location and Size
Left: 0.186"
Top: 0.19"
Width: 4.429"
Height: 0.25"
Visible? Yes
Border? No
Justify: Left
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Message Actions
Data Field: dfDescription2

Data

Maximum Data Length: Default
Data Type: String
Editable? No

Display Settings
Window Location and Size
Left: 0.186"
Top: 0.524"
Width: 4.429"
Message Actions
Data Field: dfLength
Data
  Maximum Data Length: 3
  Data Type: Number
  Editable? Yes
Display Settings
  Window Location and Size
  Left: 4.686"
  Top: 0.155"
  Width: 0.429"
  Height: 0.25"
Visible? Yes
Border? Yes
Justify: Right
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Data Field: dfScale
Data
  Maximum Data Length: 2
  Data Type: Number
  Editable? Yes
Display Settings
  Window Location and Size
  Left: 4.686"
  Top: 0.488"
  Width: 0.429"
  Height: 0.25"
Visible? No
Border? Yes
Justify: Right
Format: Unformatted
Country: Default
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default
Message Actions
Pushbutton: pbOk
Title: Ok
Window Location and Size
  Left: 5.286"
  Top: 0.155"
  Width: 1.2"
  Height: 0.298"
Visible? Yes
Keyboard Accelerator: Enter
Font Name: Default
Font Size: Default
Font Enhancement: Default
Message Actions
On SAM_Click

Verifies that an attribute's length and scale (in the case of decimals) has been declared properly.

If `dlgDetails.strCheck = 'E'
- If `dfLength < 0` or `dfLength > 15`
  - Call `SalMessageBox(Number of positions left of the decimal must be between 0 and 15.', APPNAME,MB_OK,MB_iconAsterisk)
  - Call `SalSetFocus(dfLength)`
  - Return FALSE
- If `dfScale < 0` or `dfScale > 15-dfLength`
  - Call `SalMessageBox(Both of these numbers added may not exceed 15. Number of positions right of the decimal must be between 0 and [strLength]', APPNAME,MB_OK,MB_iconAsterisk)
  - Call `SalSetFocus(dfScale)`
  - Return FALSE
- If `dfLength = 0` and `dfScale = 0`
  - Call `SalMessageBox(At least one of these values must be greater than 0.', APPNAME,MB_OK,MB_iconAsterisk)
  - Call `SalSetFocus(dfLength)`
  - Return FALSE
- Else
  - If `dfLength < 1` or `dfLength > 254`
    - Call `SalMessageBox(Lenath for a character field must be between 1 and 254.', APPNAME,MB_OK,MB_iconAsterisk)
    - Call `SalSetFocus(dfLength)`
    - Return FALSE
  - Set `dlgDetails.nLength = dfLength`
  - Set `dlgDetails.nScale = dfScale`
  - Call `SalEndDialog(hWndForm,TRUE)"

Pushbutton: pbCancel

Title: Cancel

Window Location and Size
Left: 5.286"
Top: 0.488"
Width: 1.2"
Height: 0.298"

Visible? Yes
Keyboard Accelerator: Esc

Font Name: Default
Font Size: Default
Font Enhancement: Default

Message Actions

On SAM_Click

Set `dlgDetails.nLength = dfLength`
Set `dlgDetails.nScale = dfScale`
Call `SalEndDialog(hWndForm,TRUE)"

String: strLength

Message Actions

On SAM_Create

Show length and scale (in the case of decimals) fields for user to enter.

If `dlgDetails.strCheck = 'C'` or `dlgDetails.strCheck = 'V'
- Set `dfDescription1 = 'Maximum Number Of Characters In Character Field:'`
- If `dlgDetails.nScale > -1` or `dlgDetails.nLength = -1`
  - Set `dfLength = 0`
  - Set `dlgDetails.nLength = -1`
- Else
  - Set `dfLength = dlgDetails.nLength`
  - Call `SalSetWindowText(hWndForm,'Parameters For Character Fields')`
- Else
  - Set `dfDescription1 = 'Maximum Number Of Positions Left Of The Decimal:'`
  - Set `dfDescription2 = 'Maximum Number Of Positions Right Of The Decimal:'`
  - If `dlgDetails.nScale = -1`
    - Set `dfLength = 0`
    - Set `dfScale = 0`
    - Set `dlgDetails.nLength = -1`
  - Else
Set dfLength = dlgDetails.nLength
Set dfScale = dlgDetails.nScale
Call SaISetMaxDataLength(dfLength,2)
Call SaIShowWindow(dfScale)
Call SaISetWindowText(hWndForm,"Parameters For Decimal Fields")
Set dlgDetails.bSpecificsDialog = TRUE
On SAM_Destroy
Set dlgDetails.bSpecificsDialog = FALSE
On SAM_Close
Call SaISendMsg(pbCancel,SAM_Click,0,0)

Dialog Box: dlgWait
Title: OSAM* Designer
Display Settings
Visible at Design time? No
Type of Dialog: Modeless
Window Location and Size
Left: 0.625"
Top: 0.625"
Width: 4.157"
Height: 0.905"
Absolute Screen Location? Yes
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Contents
Background Text: Loading OSAM* Designer...Please Wait.
Window Location and Size
Left: 0.75"
Top: 0.214"
Width: 3.357"
Height: 0.167"
Visible? Yes
Justify: Left
Font Name: Default
Font Size: Default
Font Enhancement: Default
Text Color: Default
Background Color: Default

Picture: plclcon
Window Location and Size
Left: 0.1"
Top: 0.083"
Width: 0.571"
Height: 0.464"
Visible? Yes
File Name: osam.lco
Storage: Internal
Fit: Size for Best Fit
Scaling
Width: 100
Height: 100
Corner: Square
Border Style: Solid
Border Thickness: 1
Tile To Parent? No
Border Color: Default
Background Color: Default

Window Variables
Message Actions
On SAM_Close
1
1 Display wait box while OSAM* Designer loads.
1
Return FALSE
Paul Francis Rabuck earned his Bachelor of Science degree from the University of North Florida in May of 1989; and expects to receive his Master of Science degree in Computer and Information Sciences from the University of North Florida in December of 1992. Dr. Susan R. Wallace is serving as Mr. Rabuck's thesis advisor.

Mr. Rabuck is a cofounder of Noe & Associates, a local software company specializing in client-server applications on PC LANs employing Microsoft Windows as a front-end. He is currently working in Columbia, South Carolina as a consultant with Noe & Associates for Strategic Data Systems, Inc., a company which specializes in property and casualty insurance computer systems. Prior to founding Noe & Associates, Mr. Rabuck worked for three years at American Surety and Casualty, a Jacksonville based insurance company, where he designed and implemented their current claims system.

Mr. Rabuck continues to work with GUIs and hopes to start working with the imaging and multimedia technologies as they relate to the user interface. His interests include music, swimming, and writing.