ODML DATABASE SCHEMA GENERATOR

Version 3.0

User Guide

Moscow, 2001

1 CONTENTS

2 INTRODUCTION

2.1 Application

The ODML Database Schema Generator is designed for developing object database schemes. A database is supported by the Object To Database Mapping Language (ODML) system library. The library provides support for an object superstructure over a classic relational database.

2.2 Requirements to User Skills

The user should have basic computer skills and study this manual. She also needs to be familiar with Pluk Language, Object To Database Mapping Language (ODML) and Structured Query Language (SQL) Reference Books. Additionally, the user should be acquainted with the Open Database Connectivity (ODBC) software that provides the standard protocol for working with Relational Database Management Systems (RDBMS's).

3 DESCRIPTION OF OPERATIONS

3.1 Main Window

The main window of the Generator displays a scheme of an object database. The scheme consists of the classes whose objects are being stored in the database. It also contains enumerations. Enumeration is a **string** type whose values are constrained by a set of strings.

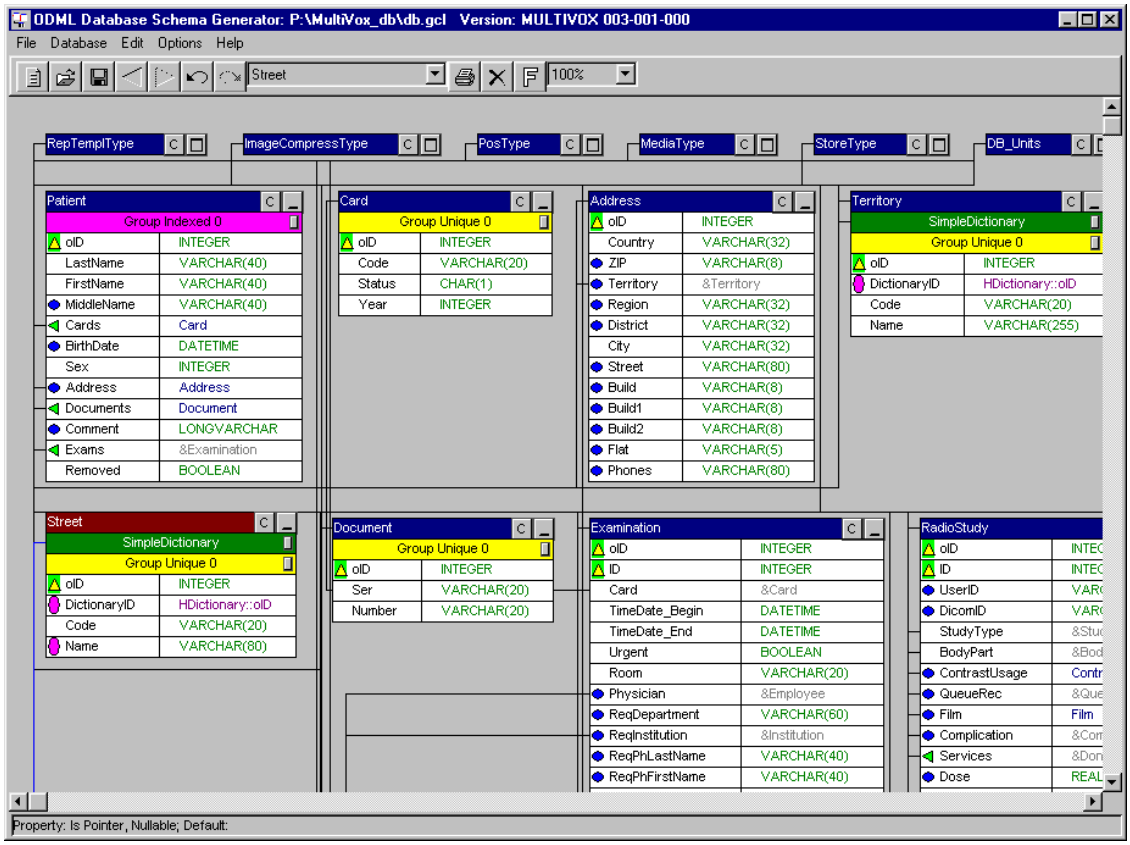

Fig. 3.1-1 Main Window

3.1.1 Main Window Menu

Fig. 3.1-2 shows the main window menu.

Fig. 3.1-2 Main Window Menu

3.1.1.1 File

This menu (Fig. 3.1-3) contains tools for working with schemes, as well as binary scheme-storing files with a *.gcl extension or text scheme-storing files with a *.cls extension. The text format for scheme storage is convenient for editing in any text editor. But this format is

used less frequently than the binary format since the latter stores additional valuable information, e.g. the tables layout in the window.

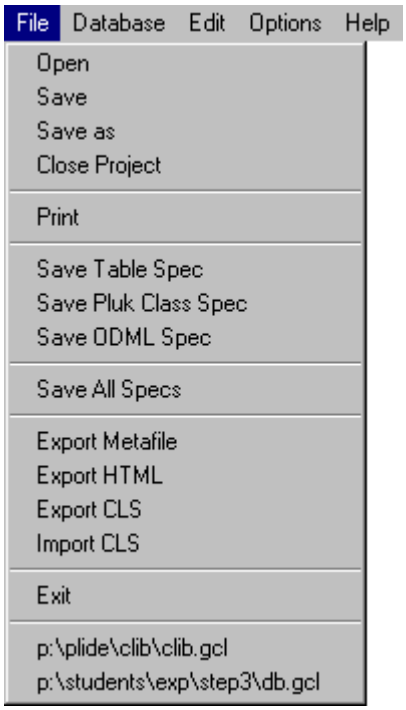

Fig. 3.1-3 *File* Menu

The last schemes opened by the user (maximum four) are shown at the menu's bottom. Selecting an item loads the corresponding scheme.

3.1.1.1.1 Open

Opens a binary scheme-storing file through the Open window for *.gcl files.

3.1.1.1.2 Save

Saves a binary scheme-storing file. If a scheme is not assigned to any file yet, the Save window for *.gcl files will open.

3.1.1.1.3 Save as

Saves a scheme into another file through the Save window for *.gcl files.

3.1.1.1.4 Close Project

Closes the current scheme. An empty scheme becomes current in which user can create new classes and enumerations.

3.1.1.1.5 Print

Prints out a scheme. The *Preview* window opens that displays the miniaturized scheme on one or several pages (Fig. 3.1-4). The user can switch between *Portrait* and *Landscape* layouts for better fit. The user can drag the scheme in a drag-and-drop mode by left-clicking on the scheme's rectangular frame. She can also proportionally resize the scheme in the same mode by left-clicking on one of the four small rectangles in the corners of the frame. If the resized scheme overlaps the space provided, additional pages will appear automatically. Similarly, the empty pages will be removed after downscaling the scheme.

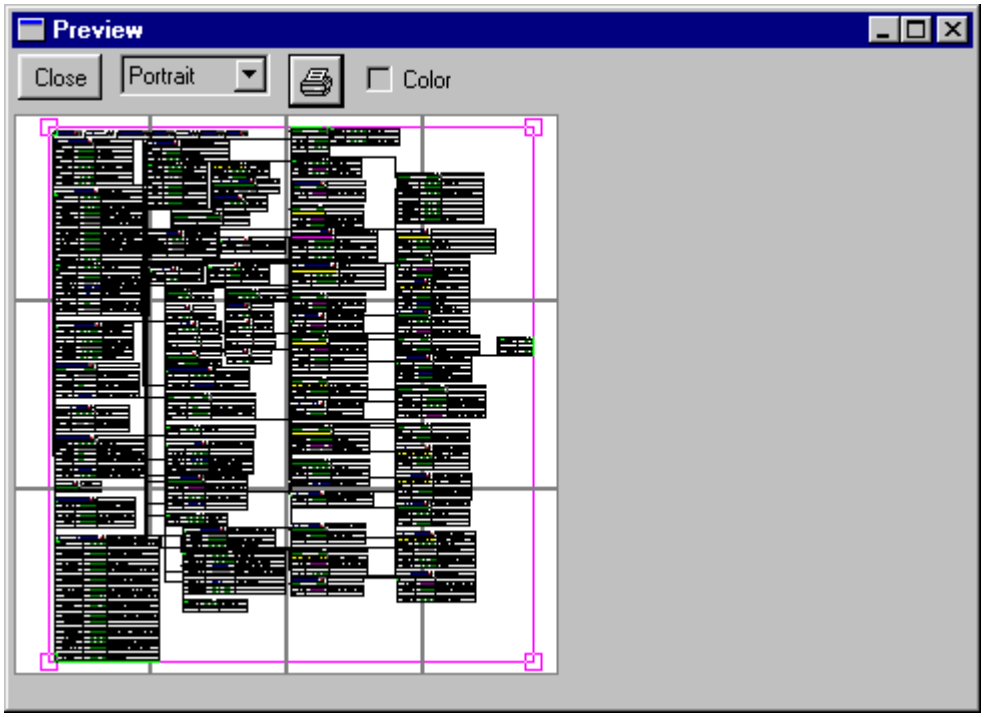

Fig. 3.1-4 *Preview* Window for Printing a Scheme

To start printing, the user must press the "printer" button. If the *Color* box is checked, the scheme will be printed in color (as seen on screen).

3.1.1.1.6 Save Table Spec

Saves the specification of the relational database scheme into which a database object scheme is mapped. This is performed through the Save window for *.qry files. The specification comprises various statements in SQL for creating tables and indexes (see SQL Reference Book).

This file is generally of no use to the user, because creation of tables and indexes can be carried out by the Generator itself (see *Create Tables*). It may be useful, however, if the user intends to apply a program that is able to execute SQL queries.

3.1.1.1.7 Save Pluk Class Spec

Saves the specification of the class scheme in Pluk Language into which a database object scheme is mapped. This is performed through the Save window for *.plk files. The specification comprises various statements in Pluk Language for creating classes, methods and vectors (see Pluk Language Reference Book). Methods are uses for access to the database. Vectors are used for storing enumeration values.

Any application able to work with database object schemes must be executed, by means of Pluk Language, the file contents.

3.1.1.1.8 Save ODML Spec

Saves the specification of a scheme of class mapping in Pluk Language into tables of the relational database. This is performed through the Save window for *.scm files. The specification is to be written in ODML (see ODML Reference Book).

Any application able to work with database object schemes must be executed, by means of ODML, the file contents.

3.1.1.1.9 Save All Specs

Saves all three specifications described in *Save Table Spec*, *Save Pluk Class Spec*, and *Save ODML Spec*. Moreover, it saves the scheme into a text file (see *Export CLS*).

3.1.1.1.10 Export Metafile

Saves a scheme into a metafile with a *.wmf extension. The *Preview* window opens displaying the miniaturized scheme on one or several pages (Fig. 3.1-5). The user can switch between *Portrait* and *Landscape* layouts for better fit. The user can drag the scheme in a drag-and-drop mode by left-clicking on the scheme's rectangular frame. She can also proportionally resize the scheme in the same mode by left-clicking on one of the four small rectangles in the corners of the frame. If the resized scheme overlaps the space provided, additional pages will appear automatically. Similarly, the empty pages will be removed after downscaling the scheme.

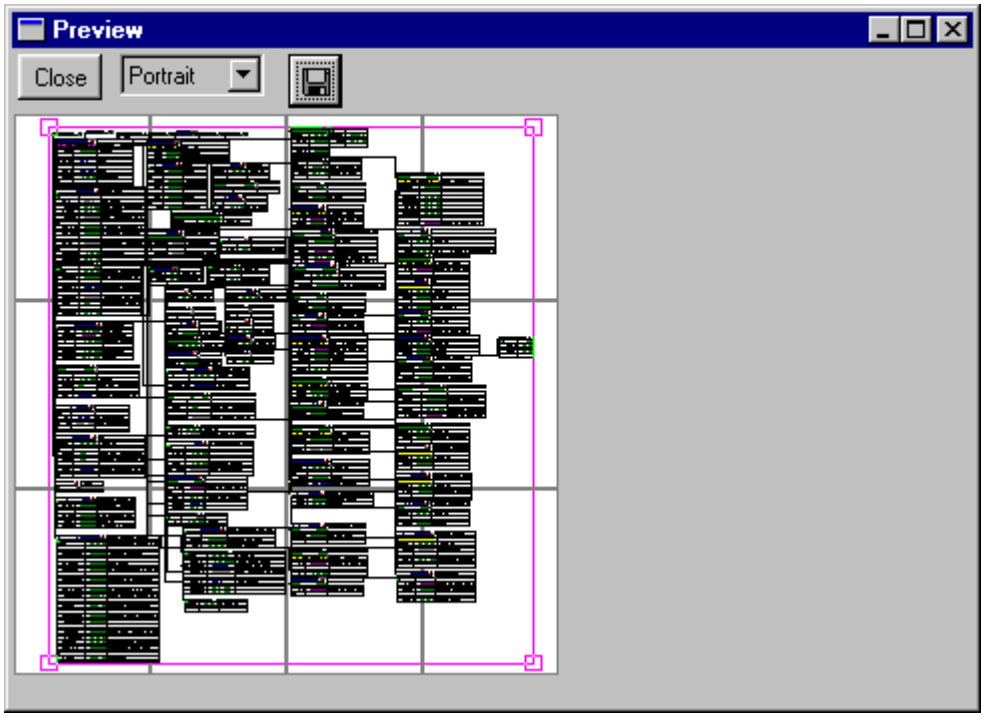

Fig. 3.1-5 *Preview* Window for Saving a Scheme into a Metafile

To save a scheme, the user must press the "floppy" button. This will be followed by opening of the Save window to enter a base name (different from any filename). Each page will be stored in a metafile under the name combined of the base name and page number. Then there will be an option whether to save in color or black-and-white format.

3.1.1.1.11 Export HTML

Saves a scheme into a hypertext file through the Save window for *.htm files.

3.1.1.1.12 Export CLS

Saves a scheme into a text file through the Save window for *.cls files.

3.1.1.1.13 Import CLS

Loads a scheme from a text file through the Open window for *.cls files.

3.1.1.1.14 Exit

Terminates the Generator session.

3.1.1.2 Database

This menu (Fig. 3.1-6) contains tools for working with databases.

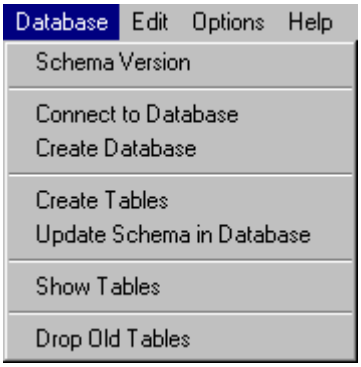

Fig. 3.1-6 *Database* Menu

3.1.1.2.1 Schema Version

Opens the *ODML Schema Version* window (see 3.2) for setting a name and version for a scheme.

3.1.1.2.2 Connect to Database

Connects to a database. First, the question *Connection to Local Data Source?* opens. *Yes*-click opens the window for selecting an ODBC source and, possibly, the source connection window for entering the username and password for access to the database. *No*-click opens the *Host Logon* window for entering the fully qualified name or IP address of the remote computer, username and password for access to the computer. Then the *Database Logon* window opens for entering the ODBC source name on the remote computer, username and password for access to the database.

3.1.1.2.3 Create Database

Creates a new ODBC source. By means of the window with the list of ODBC drivers (types of RDBMS), the user selects the needed RDBMS. Then, in a sequence of windows specific for this particular driver, the user sets up the source name, server name (for the client-server RDBMS) or file name (for the file RDBMS), etc.

3.1.1.2.4 Create Tables

Creates tables and indexes in the relational database into which a database object scheme is mapped. In order to create new tables, the user must delete old ones using *Drop Old Tables*.

3.1.1.2.5 Update Schema in Database

Updates in the database the class scheme in Pluk Language into which a database object scheme is mapped (see *Save Pluk Class Spec*) and the scheme of class mapping in Pluk Language into tables of the relational database (see *Save ODML Spec*). If a scheme of the same name and version is already being stored in the database, it will be replaced.

Schemes are stored in a special table. The existence of the table is critical for successful update. For this reason, the user needs at least one run of *Create Tables* before updating.

Schemes, being stored in the database, differ from each other by name and version and can be used by various programs that work with the database. Those programs will open a

window to select a scheme from the database, instead of using a scheme file with a *.scm extension.

3.1.1.2.6 Show Tables

Opens the *Table Browser* window (see 3.3) for viewing tables and fields of the relational database.

3.1.1.2.7 Drop Old Tables

Opens the *Drop Tables* window (see 3.4) for deleting tables from the relational database. It prompts the user to delete the tables listed in the relational scheme and ignore all other tables.

3.1.1.3 Edit

This menu (Fig. 3.1-7) contains tools for editing a scheme.

Fig. 3.1-7 *Edit* Menu

3.1.1.3.1 Undo

Cancels the last change in the scheme.

3.1.1.3.2 Redo

Restores the change canceled by *Undo*.

3.1.1.3.3 Forward

Restores the class / enumeration selection canceled by *Back*.

3.1.1.3.4 Back

Reselects the class / enumeration selected at the step before last.

3.1.1.3.5 Font

Opens the window for setting font parameters.

3.1.1.3.6 Open Comments

Shows comments on all classes / enumerations.

3.1.1.3.7 Close Comments

Closes comments on all classes / enumerations.

3.1.1.3.8 Arrange Tables

Arranges all classes / enumerations across the window.

3.1.1.3.9 Refresh

Redraws the scheme.

3.1.1.4 Options

This menu (Fig. 3.1-8) contains the Generator parameters.

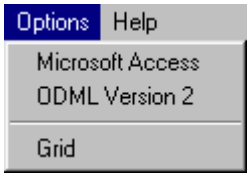

Fig. 3.1-8 *Options* Menu

3.1.1.4.1 Microsoft Access

If checked, the Generator will work with the Microsoft Access RDBMS. The Microsoft Access RDBMS specification for relational database scheme (see *Save Table Spec*) differs from the standard RDBMS specification. The reason for this lies in the fact that the Microsoft Access RDBMS does not support the following SQL elements (see SQL Reference Book):

- primary key;
- foreign key;
- field value control;
- default value of a field;
- uniqueness of a field and field group.

Because of the above, the Microsoft Access RDBMS cannot use the following elements of an object scheme:

- reference to a class attribute;
- storage of an **EMPTY** value in an attribute;
- enumeration;
- default value of an attribute;
- uniqueness of an attribute or group of attributes.

3.1.1.4.2 ODML Version 2

If this item is checked, the Generator will create the specification that describes a scheme of class mapping in Pluk Language into tables of the relational database (see *Save ODML Spec*) for the second version of ODML. As compared to version 1, version 2 supports inheritance (see 3.8) and triggers (see 3.5.2.5, 3.5.2.6, 3.5.2.7).

3.1.1.4.3 Grid

If checked, the user will be able to position the table so that the x and y coordinates of the upper left corner will have a common multiplier, i.e. this corner will always be a node of an invisible grid.

3.1.1.5 Help

The menu (Fig. 3.1-9) contains help information on the Generator.

Fig. 3.1-9 *Help* Menu

3.1.1.5.1 About

Opens the window with the version number and copyright information.

3.1.1.5.2 Help

Opens this User Guide.

3.1.2 Toolbar

3.1.2.1 

Same as *New Class* in the right-click menu of the main window (see 3.1.4.1).

3.1.2.2 

Same as *Open* in the main window menu (see 3.1.1.1.1).

3.1.2.3 

Same as *Save* in the main window menu (see 3.1.1.1.2).

3.1.2.4 

Same as *Back* in the main window menu (see 3.1.1.3.4).

3.1.2.5 

Same as *Forward* in the main window menu (see 3.1.1.3.3).

3.1.2.6 

Same as *Undo* in the main window menu (see 3.1.1.3.1).

3.1.2.7 

Same as *Redo* in the main window menu (see 3.1.1.3.2).

3.1.2.8 

The list of classes / enumerations. The item selected in the list box will be selected in the window.

If the selected class is contained in other classes or other classes are inherited from it, the list of such classes will appear in the list box at the right. If the selected enumeration is contained in other classes, the list of such classes will appear in this right list box. The item selected in the list box will be selected in the window.

3.1.2.9 

Same as *Print* in the main window menu (see 3.1.1.1.5).

3.1.2.10 ✕

Same as *Delete Class* in the class menu (see 3.5.2.11), if a class is selected, or same as *Delete Enum* in the enumeration menu (see 3.12.2.4), if an enumeration is selected.

3.1.2.11 🅵

Same as *Font* in the main window menu (see 3.1.1.3.5).

3.1.2.12 100%

Scaling tool for zooming the scheme.

3.1.3 Status Bar

Shows the properties (see 3.10.1.1) and default value (see 3.5.2.8) of the attribute pointed at by the mouse pointer.

3.1.4 Right-click Menu

Fig. 3.1-10 shows the popup menu after right-clicking off classes / enumerations.

Fig. 3.1-10 Right-click Menu

3.1.4.1 New Class

Creates a new class at the click location. First, the class name consists of the string **noname** plus a number (equaling the number of classes and enumerations in the scheme). Then the user can position this class anywhere (see 3.5), as well as change its name and comments (see 3.5.2.1).

3.1.4.2 New Enum

Creates a new enumeration at the click location. First, the class name consists of the string **noname** plus a number (equaling the number of classes and enumerations in the scheme). Then the user can position this enumeration anywhere (see 3.12), change its name and comments (see 3.12.2.1).

3.1.4.3 Font

See 3.1.1.3.5.

3.1.4.4 Refresh

See 3.1.1.3.9.

3.2 *ODML Schema Version* Window

This window (Fig. 3.2-1) is used for setting a name and version for the scheme. Schemes, being stored in the database, differ from each other by name and version and can be used by

various programs that work with the database. Those programs will open a window to select a scheme from the database, instead of using a scheme file with a *.scm extension.

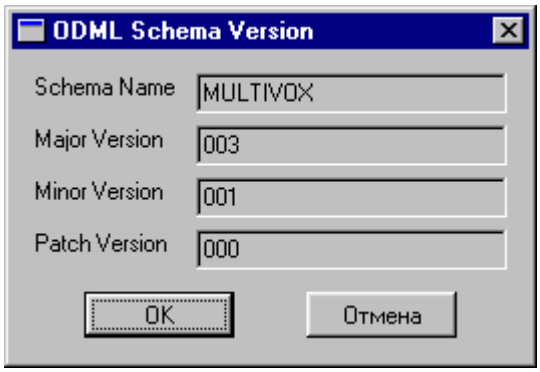

Fig. 3.2-1 *ODML Schema Version* Window

3.2.1 Schema Name

The scheme name, to tell one scheme from another in the database.

3.2.2 Major Version

A major version number, to tell the versions of one scheme in the database that considerably differ from each other.

3.2.3 Minor Version

A minor version number, to tell slightly differing versions of one scheme in the database.

3.2.4 Patch Version

A patch version number, to tell the versions of one scheme in the database that differ just in number of error corrections.

3.3 *Table Browser* Window

The user can view tables and fields of a relational database in this window (Fig. 3.3-1).

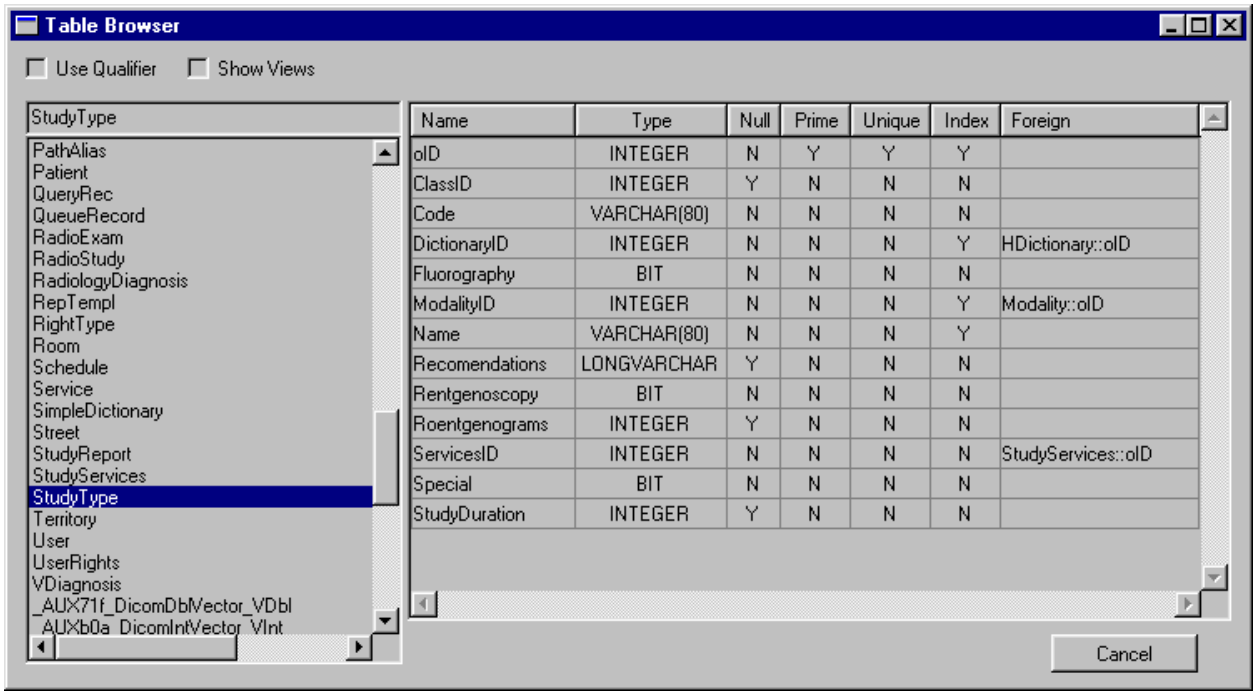

Fig. 3.3-1 ***Table Browser*** **Window**

3.3.1 Tables List

A list of relational tables of the database.

3.3.2 Fields List

A list of the fields of the table selected from the tables list. The following information is displayed for each field (see SQL Reference Book):

- name of the field;
- type in SQL notation;
- if the field can hold an **EMPTY** value;
- if the field is a primary key;
- if the field is unique;
- if the field is indexed;
- name of the foreign key.

3.3.3 Use Qualifier

If checked, the name of a table will be presented in fully qualified form: the database name, owner's name, table name (separated with periods, Fig. 3.3-2).

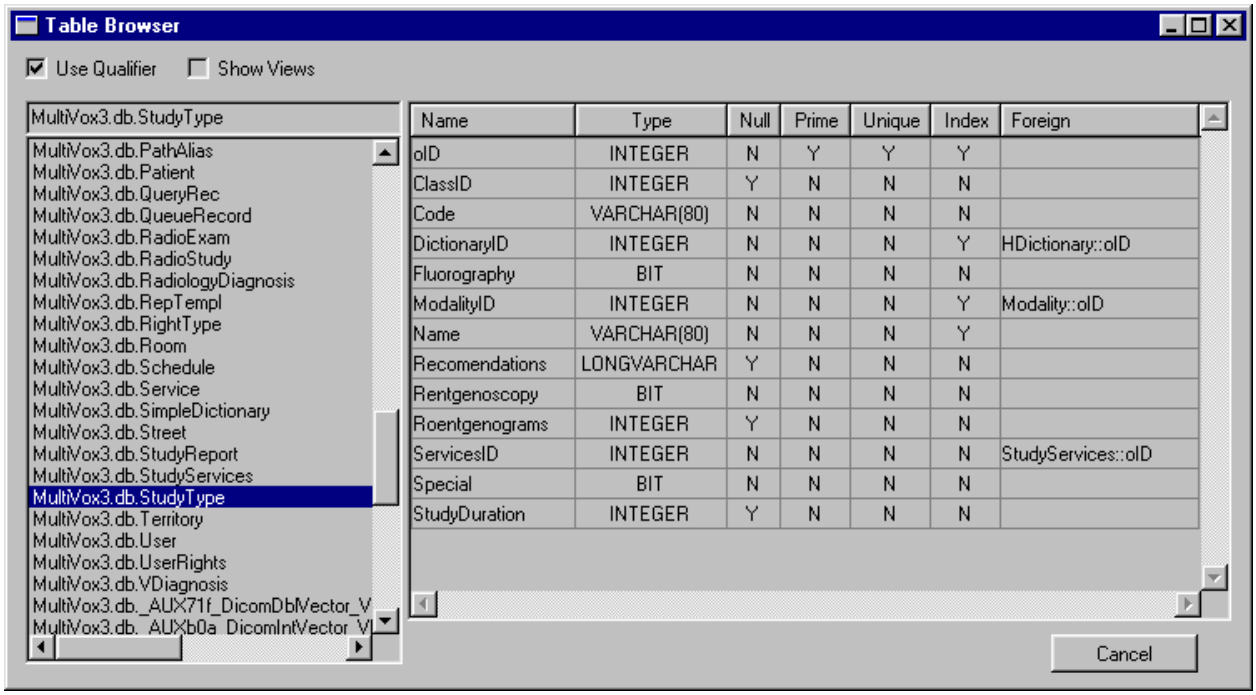

Fig. 3.3-2 *Table Browser* Window with Fully Qualified Table Names

3.3.4 Show Views

If unchecked, real tables of the database will appear on the tables list. If checked, the tables list will contain the views of the database, i.e. virtual tables obtained after a query to real tables (see SQL Reference Book).

3.4 *Drop Tables* Window

This window is used for deleting tables from the relational database (Fig. 3.4-1).

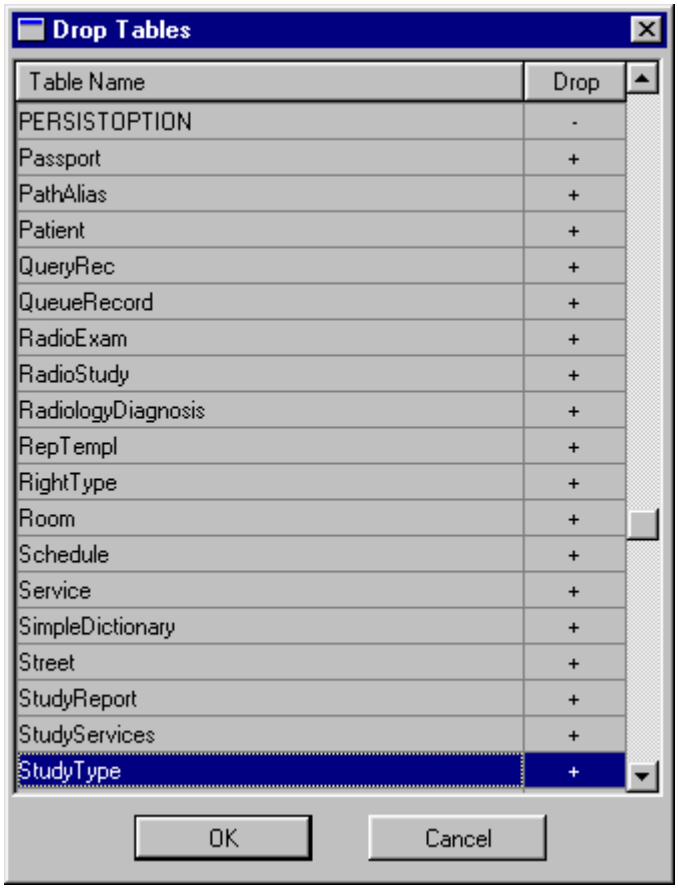

Fig. 3.4-1 *Drop Tables* Window

Tables of the relational database are listed in the *Table Name* column. In the *Drop* column, **'+'** indicates the tables to be deleted, **'–'** indicates the tables that will stay. Left double-click on an item in the *Drop* column will switch **'+'** to **'–'**, and vice versa.

3.5 Class

Each class is presented in table form (Fig. 3.5-1).

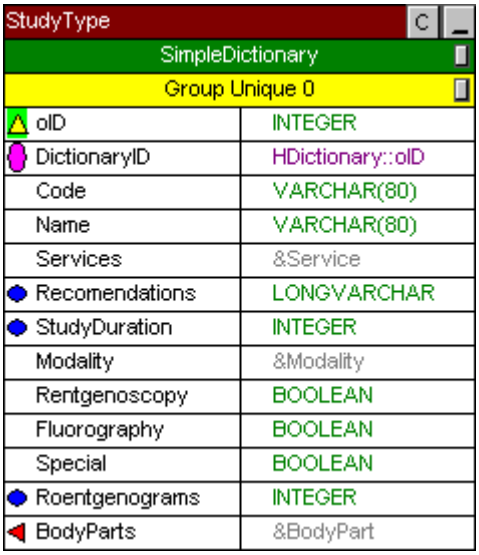

Fig. 3.5-1 Class

The name of the class is shown in the title bar. Under the title bar, on green background, there could be names of base classes (see 3.8); on pink background, names of indexing groups (see 3.9); on yellow background, names of uniqueness groups (see 3.9). Names of class attributes are presented in the left column (see 3.10). At the left of an attribute name there could be small signs indicating the attribute's properties. Types of class attributes are given in the right column. The relations of class inheritance and inclusion for the class / enumeration value in an attribute are shown with lines.

Suppose the user has selected a certain class. The color of the title bar of the selected class changes from blue to purple, and the color of the lines that depict the relations of inheritance / inclusion into the selected class from black to blue (Fig. 3.1-1). Thus one can tell which classes are contained in the selected class and from which ones it has been inherited. However, the color of the lines that depict the relation of inheritance from / inclusion of the selected class does not change. To learn how to find out which classes are inherited from the selected class and which ones contain it, see 3.1.2.8.

The user can move a class in a drag-and-drop mode.

Each class has a mandatory attribute **oID**, object identifier. The object identifier is a unique (see 3.10.1.1.4) integer with automatic numeration (see 3.10.1.1.2). The user cannot edit or delete this attribute.

3.5.1.1.1 

Shows / hides comments on the class. Fig. 3.5-2 demonstrates a class with comments. Comments at the right of the title bar relate to the class. Comments at the right of the group relate to the group, names of group attributes are listed there. Comments at the right of an attribute relate to this attribute.

StudyType		Study type
SimpleDictionary		
Group Unique 0		DictionaryID, Name, Modality
oID	INTEGER	Object ID
DictionaryID	HDictionary::oID	Root of dictionary
Code	VARCHAR(80)	Study code
Name	VARCHAR(80)	Study name
Services	&Service	Services to insurance charge
Recomendations	LONGVARCHAR	Recomendation for patient to be prepared
StudyDuration	INTEGER	Duration of study
Modality	&Modality	Modality
Rentgenoscopy	BOOLEAN	Is it x-ray scopy
Fluorography	BOOLEAN	Is it fluorography
Special	BOOLEAN	Is it special study
Roentgenograms	INTEGER	Used x-ray films for record
BodyParts	&BodyPart	Body part list

Fig. 3.5-2 Class with the Comments

3.5.1.1.2 

Minimizes the class to the title bar (Fig. 3.5-3). The relation of inheritance into / from and inclusion into the minimized class will not be shown.

Fig. 3.5-3 Minimized Class

3.5.1.1.3 

Restores the class after minimization (Fig. 3.5-1).

3.5.2 Class Menu

The class menu (Fig. 3.5-4) will pop up after the right-click on a class.

Fig. 3.5-4 Class Menu

3.5.2.1 Edit

Allows editing the class name and comments.

While editing, by pressing the ***Tab*** key the user can switch from editing the class name to editing class comments to editing the name of the first attribute, etc. Alternatively, left double-click on an item allows its editing.

3.5.2.2 Add Parent

Establishes the relation of inheritance into the class from another class. The *Add Parent* window will open for the user to choose a base class (see 3.8).

3.5.2.3 Add Index Group

Creates a group of attributes that are mutually indexed in the RDBMS. First, the group contains no elements; the user can add attributes to and delete them from the group (see 3.9).

3.5.2.4 Add Unique Group

Creates a group of attributes that are mutually unique in the RDBMS. First, the group contains no elements; the user can add attributes to and delete them from the group (see 3.9).

3.5.2.5 Trigger Load

Opens the *Method* window (see 3.6) for editing the trigger invoked upon loading an object from the database. This trigger is invoked as a class method just after loading. In case of inheritance, first, the trigger of a base class is invoked, then of a derived class.

3.5.2.6 Trigger Save

Opens the *Method* window (see 3.6) for editing the trigger invoked upon saving an object into the database. This trigger is invoked as a class method just before saving. In case of inheritance, first, the trigger of a derived class is invoked, then of a base class.

3.5.2.7 Trigger Delete

Opens the *Method* window (see 3.6) for editing the trigger invoked upon deleting an object from the database. This trigger is invoked as a class method just before deleting. In case of inheritance, first, the trigger of a derived class is invoked, then of a base class.

3.5.2.8 Set Default

Opens the *Set Default Values* window (see 3.7) for setting default values for all of the atomic non-unique attributes of the class. Default values affect only SQL queries. If, upon the insertion of a new row, the field is omitted, it will be initialized to a default value (a table field corresponds to an atomic attribute of the class).

3.5.2.9 Add Field

Adds an attribute to the class. The attribute appears at the bottom of the class-table. First, the attribute name consists of the string **field** and number of rows, and its type is **VARCHAR(80)**. Then the user can position the attribute anywhere in the table (see 3.10), change its name, type and comments (see 3.10.1.2).

3.5.2.10 Refresh Class

Redraws the class-table and lines denoting the relations with other classes.

This menu item may be helpful if the user has positioned a certain class / enumeration in the way that the relations lines should be redrawn more presentably.

3.5.2.11 Delete Class

Deletes the class.

3.6 *Method* Window

The *Method* window (Fig. 3.6-1) is used for editing triggers.

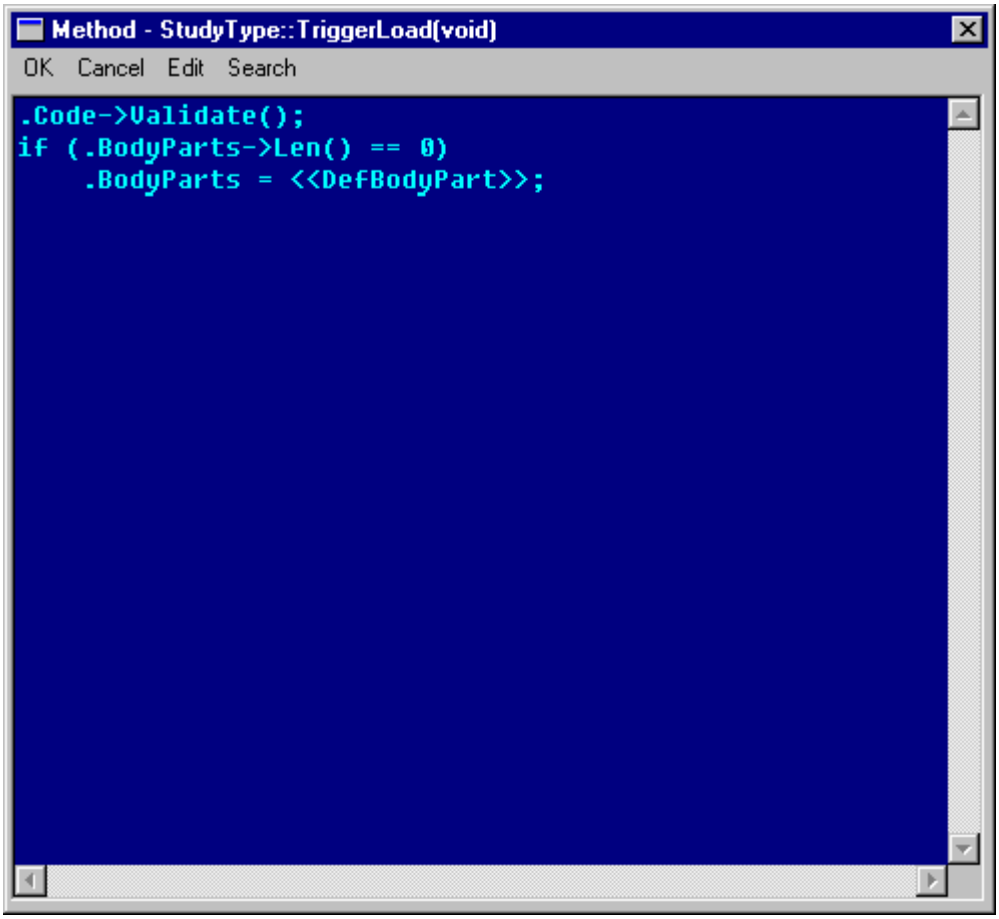

Fig. 3.6-1 *Method* Window

3.6.1 *Method* Window Menu

Fig. 3.6-2 shows the *Method* window menu.

Fig. 3.6-2 *Method* Window Menu

3.6.1.1 Edit

This menu (Fig. 3.6-3) contains tools for text editing.

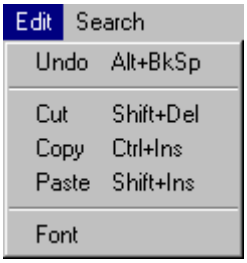

Fig. 3.6-3 *Edit* Menu

3.6.1.1.1 Undo

Cancels / restores the last change in the text being edited.

Assigned keys — ***Alt+BkSp***.

3.6.1.1.2 Cut

Copies a selected text into the clipboard, deleting it from the worksheet.
Assigned keys — ***Shift+Del***.

3.6.1.1.3 Copy

Copies a selected text into the clipboard.
Assigned keys — ***Ctrl+Ins***.

3.6.1.1.4 Paste

Inserts the clipboard contents into the cursor position.
Assigned keys — ***Shift+Ins***.

3.6.1.1.5 Font

Opens the window for setting font parameters and color.

3.6.1.2 Search

This menu (Fig. 3.6-4) contains tools for searching text.

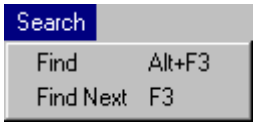

Fig. 3.6-4 *Search* Menu

3.6.1.2.1 Find

Opens the Find window.
Assigned keys — ***Alt+F3***.

3.6.1.2.2 Find Next

Searches for the next occurrence of the last entry in the text box.
Assigned key — ***F3***.

3.7 *Set Default Values* Window

The *Set Default Value* window (Fig. 3.7-1) is used for setting default values for all of the atomic non-unique attributes of a class. Default values affect only SQL queries. If, upon the insertion of a new row, the field is omitted, it will be initialized to a default value (a table field corresponds to an atomic attribute of the class).

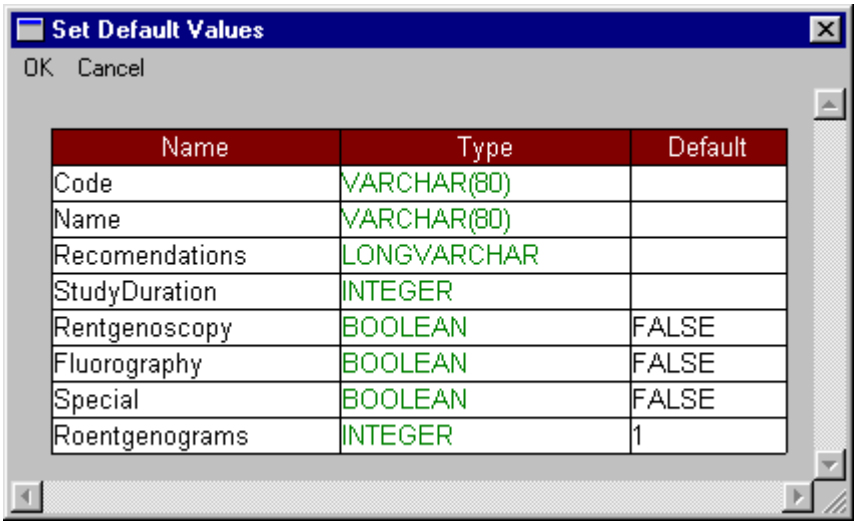

Fig. 3.7-1 ***Set Default Values*** **Window**

Attributes are presented in table form. The first (left) column contains names of the attributes. The second column contains types of the attributes. The third column contains default values of the attributes. The user can edit a default value after left-clicking on it. The ***Tab*** key allows the user to edit the next default value.

Default values are set in SQL notation (see SQL Reference Book). Note that the value of type **BOOLEAN** (not present in SQL) equals **TRUE** or **FALSE**.

3.8 Class Inheritance

The inheritance into a class (referred to as derived class) from another class (referred to as base class) allows the former to use attributes and triggers of the latter as its own. Hence, a base class is part of a derived class. Multiple inheritance is also allowed, i.e. a class can be inherited from several base classes.

The names of base classes are located under the title bar, on green background. At the right of a base class name there is a small button. If this button is depressed, the user can see the name only (Fig. 3.5-1). If the button is pressed, the user can see the open structure of the base class: its base classes, groups, attributes, and comments (Fig. 3.8-1). The structure of a base class is for viewing purpose only: the user cannot edit the base class in the table. The line that shows the relation of inheritance into the given class from the base class becomes red after left-clicking on the base class name.

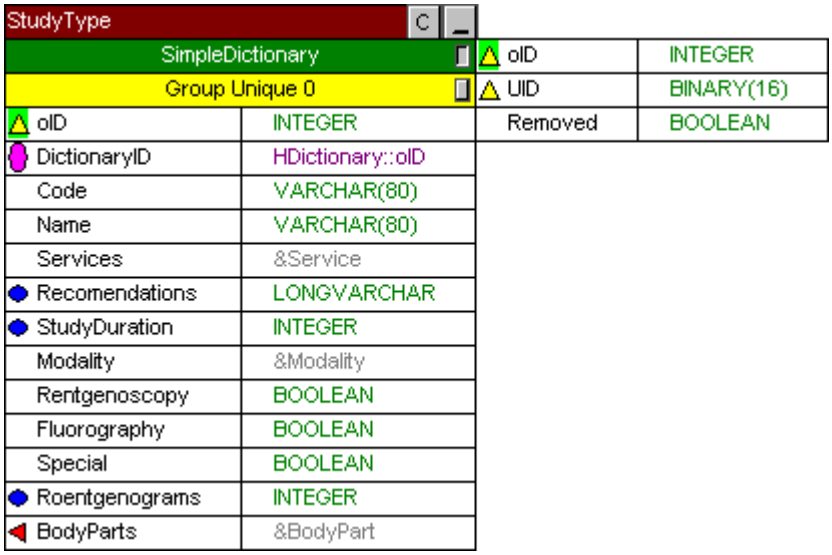

Fig. 3.8-1 Class with the Opened Structure of the Base Class

3.8.1 Base Class Menu

The base class menu (Fig. 3.8-2) pops up upon right-clicking on a base class name.

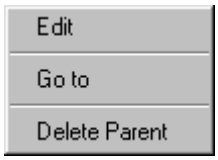

Fig. 3.8-2 Base Class Menu

3.8.1.1 Edit

Derives the class in the table from another base class. This is performed through the *Add Parent* window in which the user should select a new base class. Alternatively, this could be done by left double-clicking on the base class name.

3.8.1.2 Go to

Selects the base class.

3.8.1.3 Delete Parent

Cancels the inheritance into the class in the table from the base class.

3.9 Uniqueness and Indexing Groups

Indexing of class objects by an attribute group can considerably expedite access for a query that uses some of these attributes in the clause **where** (see ODML and SQL Reference Books). If there is a combination of attributes from different groups (or attributes of one group and non-indexed attributes) in the clause, the possibility and degree of expedition will depend on the type of RDBMS.

Uniqueness of class objects by an attribute group means that there cannot be two objects for which the values of the attributes of this group are equal.

Names of indexing groups are shown under the title bar on pink background; uniqueness groups names are shown on yellow background. Attributes of the group are listed in group comments. At the right of the group name there is a small button. If the button is pressed, the grouped attributes are highlighted pink / yellow in the table (Fig. 3.9-1). The attribute's number in the group is shown at the left of the attribute (overlaying the attribute's property sign).

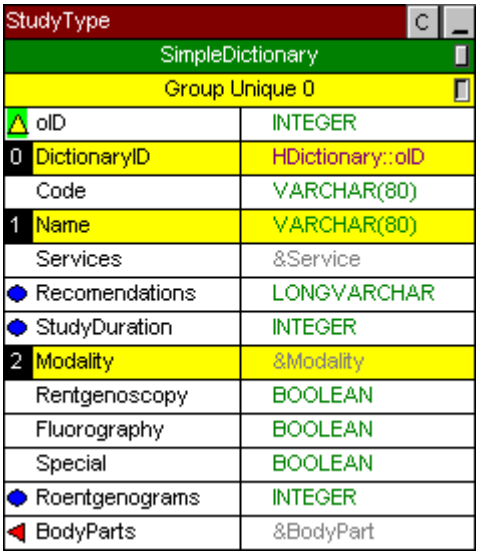

Fig. 3.9-1 Class with Highlighted Grouped Attributes

3.9.1 Group Menu

The group menu (Fig. 3.9-2) pops up upon right-clicking on a group name.

Fig. 3.9-2 Group Menu

3.9.1.1 Set Order

Allows the user to set an order for attributes in the group. Selection of this item highlights all the attributes of the group (the small button at the right of the group name will be pressed automatically, Fig. 3.9-1). Then the user sets up the order by clicking on the attributes one by one (save the last one).

3.9.1.2 Delete Group

Deletes the group removing all attributes from it.

3.10 Class Attribute

Each class attribute has the name, type and properties.
Following are the classes of the attribute types and corresponding colors.

Type	Color	Description
Atomic	Green	Attribute contains a value of atomic (ultimate) type supported by SQL (see SQL Reference Book); in addition to SQL, the types **BOOLEAN, DATE,**

		TIME are supported
Class / Enumeration	Blue	Attribute contains an object of a class / enumeration
Pointer to Class	Gray	Attribute contains a pointer to a class object
Reference to Class Attribute	Pink	Attribute contains a reference to a unique atomic class attribute
Nonexistent	Red	

Following are the atomic types and ranges of their values for working with the ODBC driver for the Microsoft SQL RDBMS.

Type	Value
CHAR(N)	N characters
BOOLEAN	TRUE or FALSE
BINARY(N)	N bytes
DATETIME	Date and time
DOUBLE	Double precision floating-point integer (from $-1.79 \cdot 10^{308}$ to $1.79 \cdot 10^{308}$)
INTEGER	Integer (from -2^{31} to $2^{31}-1$)
SMALLINT	Short integer (from -2^{15} to $2^{15}-1$)
TINYINT	Byte (from 0 to 255)
REAL	Floating-point integer (from $-3.4 \cdot 10^{38}$ to $3.4 \cdot 10^{38}$)
FLOAT	Double precision floating-point integer (from $-1.79 \cdot 10^{308}$ to $1.79 \cdot 10^{308}$)
LONGVARCHAR	Any number of characters
LONGVARBINARY	Any number of bytes
DATE	Date
TIME	Time of day
TIMESTAMP	A unique identifier in the database that changes with any change in a object
VARCHAR(N)	No more than N characters
VARBINARY(N)	No more than N bytes

Number of characters / bytes in a value of the types **CHAR, BINARY, VARCHAR, VARBINARY** is limited to eight kilobytes. For storing longer strings, one needs to use the types **LONGVARCHAR, LONGVARBINARY** which have a drawback: occupy the RDBMS memory whose size is divisible by four (kilobytes). The type **DATETIME** cannot contain a date preceding January 1, 1753. The types **DATE** and **TIME** are not altogether supported by the Microsoft SQL RDBMS (although supported by other RDBMS's).

A small sign can appear at the left of an attribute. It corresponds to a particular property of the attribute. If the attribute has more than one property, the signs cover each other. Following is the list of attribute properties and their corresponding signs:

Sign	Color	Description
	Green	Attribute with automatic numeration (see 3.10.1.1.2)
	Pink	Indexed attribute (see 3.10.1.1.3)
	Yellow	Unique attribute (see 3.10.1.1.4)
	Blue	Attribute can hold an **EMPTY** value (see 3.10.1.1.5)
	Green	Vector of the values of a given type (see 3.10.1.1.6)
	Red	Sorted vector of the values of a given type (see 3.10.1.1.7)

The user can move an attribute within a class in a drag-and-drop mode while holding down the ***Ctrl*** key.

The user can copy attributes from one class into another. The selection is performed by left-clicking on the attributes while holding down the ***Shift*** key. The selected rows turn gray. (Fig. 3.10-1). The user can deselect a selected attribute by left-clicking on it while holding down the ***Shift*** key.

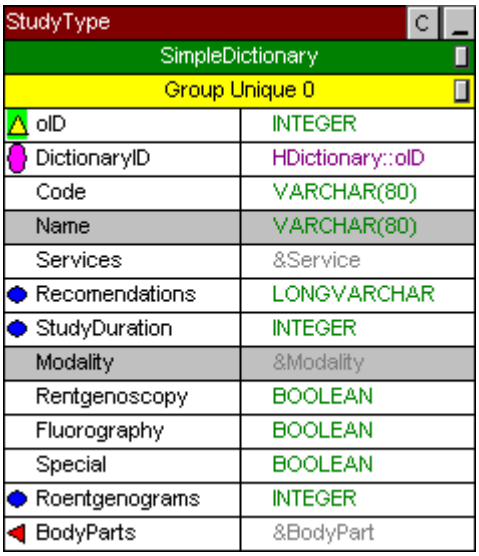

Fig. 3.10-1 Class with the Attributes Selected for Copying

Upon selection of attributes, the *Paste* menu will appear in the class menu (Fig. 3.10-2) and attribute menu (Fig. 3.10-3).

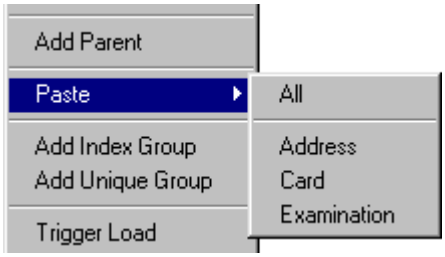

Fig. 3.10-2 Class Menu Fragment with the *Paste* Menu

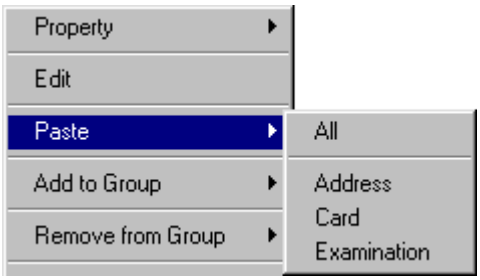

Fig. 3.10-3 Attribute Menu Fragment with the *Paste* Menu

If pasted via the class menu, the copied attributes will be inserted at the bottom of the class-table. If pasted via the attribute menu, the copied attributes will be inserted above this attribute. The item *All* in the *Paste* menu allows inserting attributes selected in all classes. The item with a class name in the *Paste* menu allows inserting the selected attributes of this class only.

3.10.1 Attribute Menu

The attribute menu (Fig. 3.10-4) pops up upon right-clicking on an attribute.

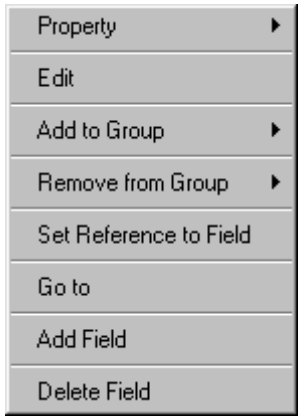

Fig. 3.10-4 Attribute Menu

3.10.1.1 Property

Each item of this menu (Fig. 3.10-5) can be checked / unchecked by the user assigning / removing the property.

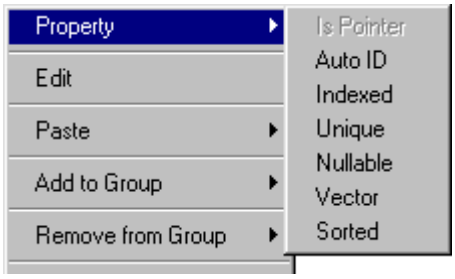

Fig. 3.10-5 *Property* Menu

3.10.1.1.1 Is Pointer

If this item is checked, the attribute contains a pointer to an object of a certain class. In this case, the type will be preceded by an **&**-sign and its color will be gray.

This item can be checked only in an attribute of type **class**.

3.10.1.1.2 Auto ID

If this item is checked, the attribute supports automatic numeration. If an object is created in the database and the value of the attribute with automatic numeration equals **EMPTY**, this value will be set up automatically, equaling the maximum value of this attribute among all of the existing objects of the class plus one.

This item can be checked only in an attribute of atomic type. As of today, only numeration of integral types has been implemented.

3.10.1.1.3 Indexed

If this item is checked, the attribute is indexed. Indexing of class objects by an attribute considerably expedite access for a query that uses this attribute in the clause **where** (see ODML and SQL Reference Books). If there is a combination of this attribute and non-indexed attributes in the clause, the possibility and degree of expedition will depend on the type of RDBMS.

This item cannot be checked in an attribute of type **vector**.

3.10.1.1.4 Unique

If this item is checked, the attribute is unique. Uniqueness of class objects by an attribute means that there cannot be two objects in which the values of the attribute are equal.

This item cannot be checked in an attribute of type **vector**.

3.10.1.1.5 Nullable

If this item is checked, the attribute can hold an **EMPTY** value.

This item cannot be checked in an attribute of type **vector**.

3.10.1.1.6 Vector

If this item is checked, the attribute is a vector of values of the given type.

3.10.1.1.7 Sorted

If this item is checked, the attribute is a vector of values of the given type arranged in the order of their entry into the database.

3.10.1.2 Edit

Allows editing the name, type and comments on the attribute.

While editing, by pressing the ***Tab*** key the user can switch from editing the attribute name to editing the attribute type to editing comments on the attribute to editing the name of the next attribute, etc. Alternatively, left double-click on an item allows its editing. The line that shows the relation of inclusion into the attribute being edited is red.

3.10.1.3 Add to Group

This menu contains names of indexing and uniqueness groups (see 3.9). By selecting an item the user adds the attribute to the selected group.

3.10.1.4 Remove from Group

This menu contains names of indexing and uniqueness groups (see 3.9). By selecting an item the user removes the attribute from the selected group.

3.10.1.5 Set Reference to Field

Opens the *Select Reference to Field* window (see 3.11) for setting up a reference of the attribute to a unique atomic attribute of some class. In this case, the type of the attribute is shown as follows: **class::field**, where **class** is the name of the class being referred to by the given attribute, **field** is the name of the attribute being referred to by the given attribute.

3.10.1.6 Go to

Selects the class if the attribute is of type **class**, pointer to object, or reference to class attribute. Selects the enumeration, if the attribute is of type **enumeration**.

3.10.1.7 Add Field

Adds a new attribute to the class. In the class-table, the new attribute will be positioned above the given attribute. First, its name consists of the string **field** plus number of rows in the table, and its type is **VARCHAR(80)**. Then the user can position the attribute anywhere in the table (see 3.10), as well as change its name, type and comments (see *Edit*).

3.10.1.8 Delete Field

Removes the attribute from the class.

3.11 *Select Reference to Field* Window

This window (Fig. 3.11-1) is used for setting up a reference of the attribute to a unique atomic attribute of some class.

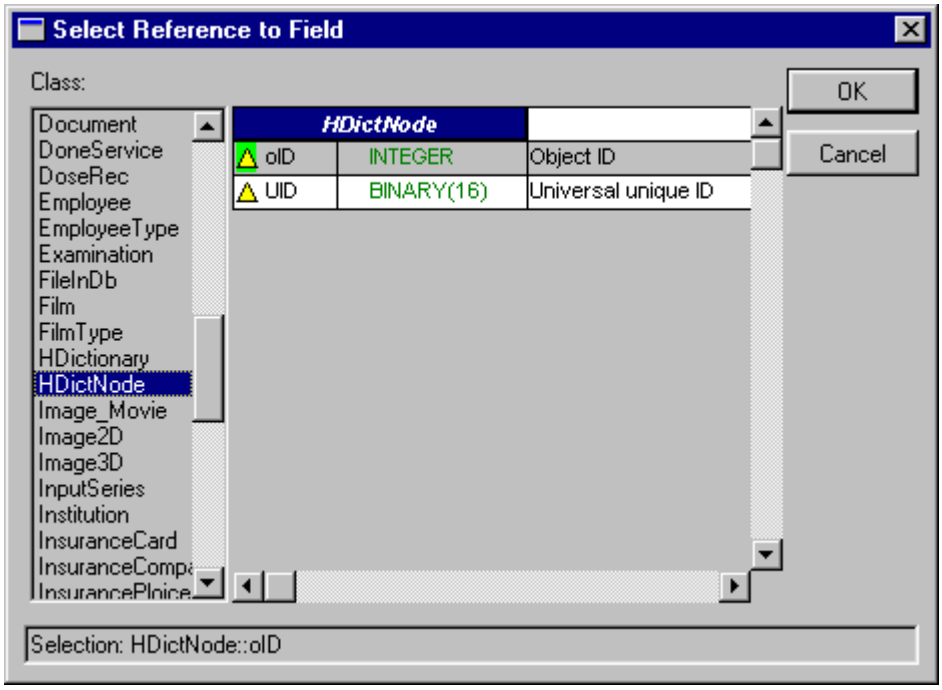

Fig. 3.11-1 *Select Reference to Field* Window

First, the user must select a class from the *Class* list. At the right of the list there is the table with names, types and comments of unique atomic attributes of the selected class. The user should select an attribute to be referred to. The selected attribute will turn gray. The type of the reference will be shown at the bottom as follows: **class::field**, where **class** is the name of the class being referred to, **field** is the name of the attribute being referred to.

3.12 Enumeration

Enumeration is a **string** type whose values are constrained by a set of strings. Enumeration are presented in table form (Fig. 3.12-1).

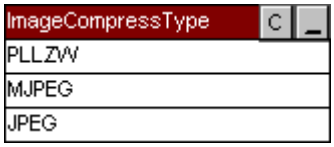

Fig. 3.12-1 Enumeration

The name of the enumeration is shown in the title bar. Under the title bar, there are rows with enumeration values (see 3.13). The relation of inclusion of the enumeration into an attribute is depicted by a line.

If the user selects an enumeration, the color of its title bar will change from blue to purple. To find out which classes contain the enumeration, see 3.1.2.8.

The user can move the enumeration in a drag-and-drop mode.

3.12.1.1.1 

Shows / hides comments on an enumeration. Fig. 3.12-2 shows an enumeration with comments.

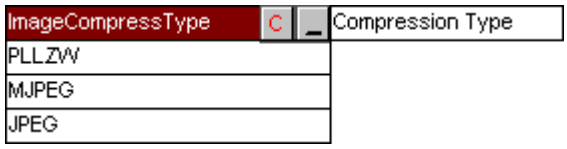

Fig. 3.12-2 Enumeration with Comments

3.12.1.1.2 

Minimizes the enumeration to the title bar (Fig. 3.12-3).

Fig. 3.12-3 Minimized Enumeration

3.12.1.1.3 

Restores the enumeration after minimization (Fig. 3.12-1).

3.12.2 Enumeration Menu

The enumeration menu (Fig. 3.12-4) pops up upon right-clicking on an enumeration.

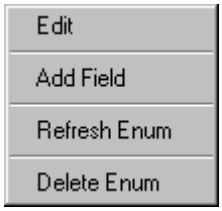

Fig. 3.12-4 Enumeration Menu

3.12.2.1 Edit

Allows editing the enumeration name and comments.

While editing, by pressing the ***Tab*** key the user switches from editing the enumeration name to editing comments to editing the value of the first string, etc. Alternatively, left double-click on an item allows its editing.

3.12.2.2 Add Field

Adds a new string to the enumeration. The string (row) appears at the bottom of the enumeration-table. First, its value equals **string** plus number of rows. Then the user can position it anywhere in the table (see 3.13) and change its value (see 3.13.1.1).

3.12.2.3 Refresh Enum

Redraws the enumeration-table and lines of the relations with other classes.

This menu item may be helpful if the user has positioned a certain class / enumeration so that the relation lines should be redrawn more presentably.

3.12.2.4 Delete Enum

Deletes the enumeration.

3.13 Enumeration String

The user can move a string within an enumeration in a drag-and-drop mode while holding down the ***Ctrl*** key.

The user can copy strings from one enumeration into another. The selection is performed by left-clicking on the strings while holding down the ***Shift*** key. The selected rows turn gray (Fig. 3.13-1). The user can deselect a selected string by left-clicking on it while holding down the ***Shift*** key.

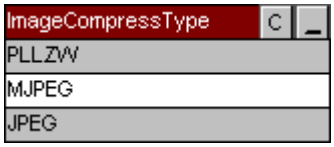

Fig. 3.13-1 Enumeration with the Strings Selected for Copying

Upon selection of the strings, the *Paste* menu will appear in the enumeration menu (Fig. 3.13-2) and enumeration string menu (Fig. 3.13-3).

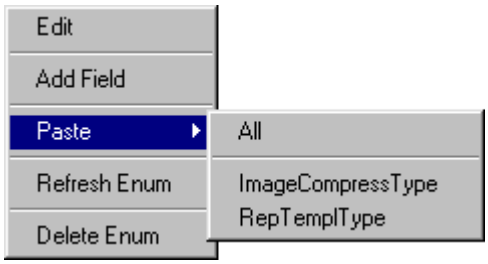

Fig. 3.13-2 Fragment of the Enumeration Menu with the *Paste* Menu

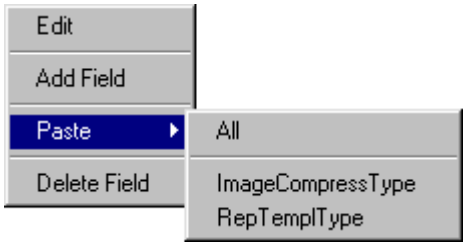

Fig. 3.13-3 Fragment of the Enumeration String Menu with the *Paste* Menu

If pasted via the enumeration menu, the copied strings will be inserted at the bottom of the enumeration-table. If pasted via the enumeration string menu, the copied strings will be inserted above this string. The item *All* in the *Paste* menu allows inserting selected strings from all enumerations. The item with an enumeration name in the *Paste* menu allows inserting the selected strings from this enumeration only.

3.13.1 Enumeration String Menu

The enumeration string menu (Fig. 3.13-4) pops up upon right-clicking on a string.

Fig. 3.13-4 Enumeration String Menu

3.13.1.1 Edit

Allows editing the string.

While editing, by pressing the ***Tab*** key the user can switch to editing the next string, etc. Alternatively, left double-click on the string allows the user to edit it.

3.13.1.2 Add Field

Adds a new string to the enumeration. In the enumeration-table, the new string will be positioned above the given string. First, its value equals **string** plus number of rows in the table. Then the user can position it anywhere in the table (see 3.13) and change its value (see *Edit*).

3.13.1.3 Delete Field

Removes the string from the enumeration.