

CUSTOMIZATION TOOL USER'S GUIDE

May 15, 2016, Release 1.2, web version

This document contains information which is the property of Dialogue Technologies. All rights are reserved. Dialogue Technologies does not guarantee or in any way represent that the information is accurate and/or complete and assumes no responsibility for any errors, omissions or other issues with the information. The information in the document is provided AS IS. Dialogue Technologies does not assume any responsibility for any consequences of using the information contained herein.

CUSTOMIZATION TOOL USER'S GUIDE

Second Edition (May 2016)

This major revision obsoletes and replaces all previous versions. This edition applies to Release 2.0 of Ergo, and to all subsequent releases and modifications until otherwise indicated in new editions. Address any comments you may have on the document to:

Dialogue Technologies AB
Ankdammsgatan 20
S-171 43 Solna, Sweden, or
info@dialoguetech.com

When you send information to Dialogue Technologies, you grant Dialogue Technologies a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright Dialogue Technologies AB 2016. All rights reserved.

CUSTOMIZATION TOOL USER'S GUIDE

CONTENTS

1	Ergo components	10
1.1	The query interface	11
1.2	The natural language engine	13
1.3	The Customization Tool	14
1.3.1	Basic concepts.....	14
2	Planning and preparation	17
2.1	Planning a Ergo project.....	17
2.2	Determining the scope of the project	17
2.3	Business factors.....	18
2.4	Database factors	18
2.5	Pre-customizing	19
2.6	Customizing	19
2.7	Model testing.....	19
2.8	Maintaining the model.....	20
3	Getting started.....	21
3.1	Installing or launching the Customization Tool component	21
3.2	Extracting database catalog information.....	21
3.3	Starting the Customization Tool program	21
4	Working with the database.....	22
4.1	Using Prolog predicates	22
4.2	Using SQL to define the database	23
4.3	Adding tables to the conceptual model.....	23
4.4	Working with the database diagram.....	25
4.5	Hiding columns.....	26
4.6	Defining column information.....	27
4.7	Defining keys.....	28
4.8	Defining case information.....	28
4.9	Excluding columns	29
4.10	Defining dependencies.....	29
4.11	Defining join paths.....	31
4.12	Defining inclusion dependencies	31
4.13	Moving between database mode and language mode	33
5	Defining table and column entities	34
5.1	Defining, naming, and identifying table entities	34
5.2	Naming entities	35
5.3	Direct and indirect table entities.....	35
5.4	Classifying names and identifiers	35
5.5	Defining terms for names and identifiers	36
5.6	Defining name and identifier relationships.....	36

CUSTOMIZATION TOOL USER'S GUIDE

5.7	Defining refer-to relationships	37
5.8	Processing the other column entities	38
5.9	Classifying entities	38
5.9.1	Thing class.....	40
5.9.2	Locative classes.....	40
5.9.3	Temporal classes.....	40
5.9.4	Numerical classes.....	40
5.10	Specifying terms for entities	42
5.10.1	Types of compound terms	42
5.11	Defining relationships.....	43
5.11.1	Possessive relationships	44
5.11.2	Locative (place) relationships	45
5.11.3	Location words.....	46
5.11.4	Temporal (time) relationships	47
5.11.5	Time words	48
5.11.6	Duration relationship.....	49
5.11.7	Measured-by relationship	49
5.12	Prepositions between nouns	51
6	Defining subsets of the data.....	53
6.1	Creating noun subclass entities	53
6.2	Creating instance entities.....	54
6.3	Creating adjective subclasses.....	55
6.4	Implicit adjectives	55
6.5	Specific adjectives.....	56
6.6	General adjectives	57
6.7	Relationships with adjective entities	57
6.7.1	Locative relationships	57
6.7.2	Temporal relationships	58
6.7.3	Prepositional relationships.....	58
7	Defining multicolumn entities	60
7.1	Calculated entities.....	60
7.1.1	Handling null values.....	61
7.2	Composite entities.....	61
7.3	Composite identifiers.....	62
7.4	Composite names	63
7.5	Composite columns.....	63
7.6	Composite reports.....	64
7.7	Associated relationships	64
8	Defining other indirect entities.....	67
8.1	Creating verbs.....	67
8.2	Defining the verb complement	67

CUSTOMIZATION TOOL USER'S GUIDE

8.3	Specifying prepositional complements.....	68
8.3.1	Implicit prepositions	68
8.3.2	Explicit prepositions	68
8.3.3	Verbs with particles.....	69
8.4	Defining verb relationships.....	70
8.4.1	Subject/object relationships	70
8.4.2	Causal relationships	70
8.4.3	Relationships of manner	71
8.4.4	Implicit prepositions	71
8.4.5	Explicit prepositions	71
8.5	Verbs in passive voice	72
8.6	Entities referring to numerical data	72
8.6.1	Counted-by relationships	72
8.6.2	Unit entities	74
8.6.3	Quantified_property and unit_of_measure entities.....	74
9	Testing the conceptual model	75
9.1	Making the conceptual model available	75
9.2	Asking questions	75
9.3	Questions that use only nouns.....	75
9.4	Questions that use verbs	76
9.5	Questions that use adjectives	76
9.6	Getting information about your model.....	76
9.7	Dealing with problems.....	76
9.8	Making the model available to users.....	77
10	Managing the model.....	78
10.1	File.....	78
10.2	New	79
10.3	Open.....	79
10.4	Properties	79
10.5	Save	80
10.6	Save as	80
10.7	Print	81
10.8	Exit	81
10.9	Edit	81
10.10	Undo.....	82
10.11	Copy	82
10.12	Paste	82
10.13	Clear	82
10.14	Move.....	83
10.15	Find	84
10.16	Action	84

CUSTOMIZATION TOOL USER'S GUIDE

10.17 Add tables	85
10.18 Collapse table	85
10.19 Hide columns	86
10.20 Create entity	86
10.21 Cluster	87
10.22 Expand	89
10.23 Define	89
10.24 Rotate	90
10.25 Snap to grid	90
10.26 Mode	90
10.27 Database	90
10.28 Language	90
10.29 View	91
10.30 Scale diagram	91
10.31 Fit diagram	92
10.32 Overview	92
10.33 Center diagram	93
10.34 Refresh	93
10.35 Options	94
10.36 Styles	94
10.37 Show labels	98
10.38 Show information area	99
10.39 Grid	99
10.40 Name break	100
10.41 Preferences	101
11 Specifying entities	104
11.1 The language diagram	104
11.2 Working with the Define entity window	105
11.3 Naming an entity	107
11.4 Classifying an entity	108
11.5 Subclass and instance entities	108
11.6 Composite entity	111
11.7 Intersecting entities	112
11.8 Units of measure	113
11.8.1 Selecting an existing unit of measure	114
11.9 Changing or deleting an existing classification	114
11.10 Creating terms	115
11.10.1 Noun grammar	117
11.10.2 Verb grammar	119
11.11 Specifying syntax	120
11.12 Syntax for nouns and adjectives	121

CUSTOMIZATION TOOL USER'S GUIDE

11.13 Verb syntax	123
11.14 Writing SQL statements	126
11.15 Handling multiple entities	127
12 Specifying relationships	130
12.1 How relationships are shown in the diagram	130
12.2 Creating or changing a relationship	130
12.3 Removing a relationship.....	132
12.4 Correcting invalid relationships	132
13 Extracting database catalog information	133
14 Producing reports	134

Appendices

CUSTOMIZATION TOOL USER'S GUIDE

Notices

References in this publication to Dialogue Technologies products, programs, or services do not imply that Dialogue Technologies intends to make these available in all countries in which Dialogue Technologies operates. Any reference to a Dialogue Technologies product, program, or service is not intended to state or imply that only Dialogue Technologies' product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of Dialogue Technologies' intellectual property rights may be used instead of the Dialogue Technologies product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by Dialogue Technologies are the user's responsibility.

Dialogue Technologies may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the Dialogue Technologies AB, Ankdammsgatan, [171 43 Solna](#), Sweden.

Trademarks

The following terms in this publication are trademarks of other companies as follows:

DB2	IBM
Structured Query Language/Data Services	IBM
SQL/DS	IBM

About this book

This book describes how to use the Customization Tool of Dialogue Technologies' Ergo products, referred to in this book as Ergo. A good understanding of Structured Query Language (SQL) is helpful. A detailed understanding of linguistics is not required, although a basic understanding of school grammar is a definite advantage.

You should also be familiar with the host system where the natural language engine resides, e.g. Unix, Linux, Android or Windows.

Terminology

In Ergo, the basic element in a conceptual model, from the language perspective, is an entity. An entity can have a direct or an indirect relationship to any table or column in the database. In entity-relationship (ER) data modeling techniques, which focus on the database rather than the language perspective of data structures, the word entity generally refers to tables only.

About the examples

The examples in this book use the EPE sample database and the EPEDEMOE sample conceptual model that are supplied with Ergo. Use the samples to explore the functions of the customization tool, and to see what a developed conceptual model can look like.

See Appendix B, "Sample tables" on page for information about the contents and structure of the EPE sample database.

Part 1. Overview

1 ERGO COMPONENTS

Ergo is a product that lets users retrieve data from database management systems (DBMS) or from other data repositories by asking questions in their own words, using natural language. Ergo can also be used to issue commands in natural language to control products or services. This chapter provides an overview of the main components of Ergo and of their relationships to the database and the presentation facility.

There are two main categories of Ergo applications namely, corpus-centric and database-centric applications. The former refer to e.g. customer self-service applications where the questions asked by customers/consumers make up the application domain, universe of discourse (UOD), and the task is to build an application to answer these questions, or issue these commands. The latter refer to applications where you want to make the contents of a database available to users. A typical example is a yellow pages self-service application where users can ask for addresses, phone numbers, etc. in natural language. In this case the application domain is determined by the contents in the database.

Most of the examples in this book will be taken from a database-centric application in which users can ask natural language questions to a database containing information about a company.

The components of Ergo are:

- Query interface provides the general user with the means of asking natural language questions, of retrieving data from the database, and of seeing the results of questions through a presentation facility. Dialogue Technologies today supports a wide variety of query interfaces, including voice-, web-, mail and text interfaces.
- Natural language engine analyzes questions put by users in a natural language, and generates the SQL statements needed to retrieve data from the database.
- Customization tool is used to create a mapping of words and relationships in the natural language questions to the tables and columns of the database. This mapping is defined in a conceptual model developed by the customizer and is used by the natural language engine when analyzing natural language questions.

The natural language engine in Dialogue Technologies Ergo comes with an Application Programming Interface (API). The programming interface let Dialogue Technologies, customers, and vendors write query interfaces between the general user and the natural language engine.

Figure 1 gives an overview of the Ergo components and of their relationship to the database and presentation facilities.

CUSTOMIZATION TOOL USER'S GUIDE

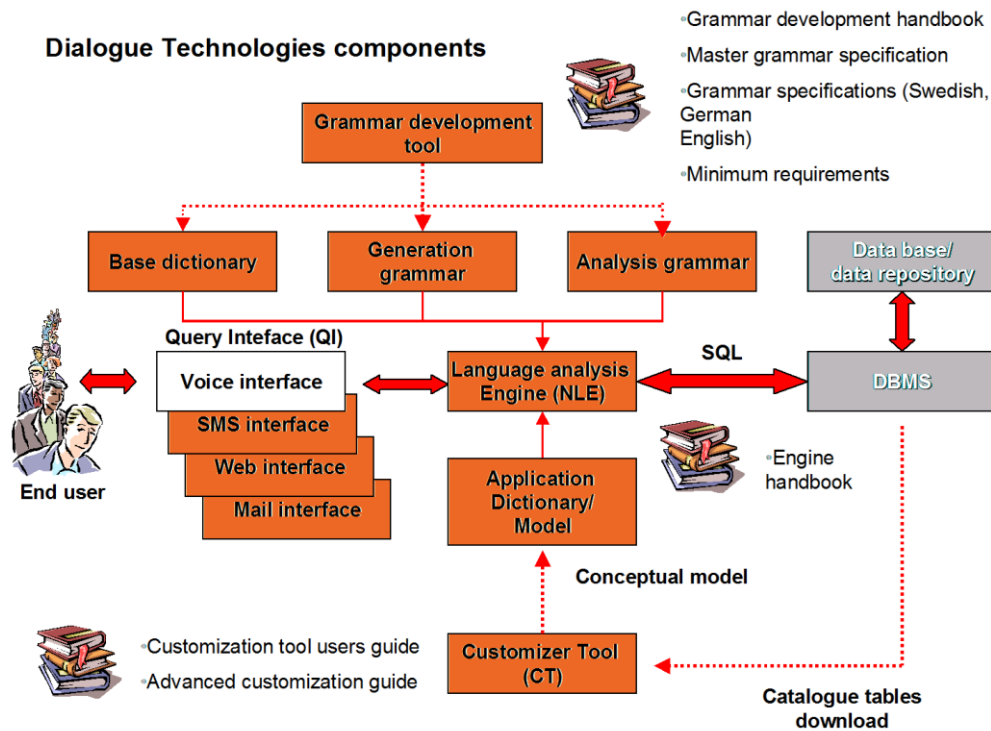


Figure 1. Ergo components.

The natural language engine is available for Unix, Linux, Android and Windows platforms. Specific system requirements are listed in a Systems Requirements document.

The Customization Tool is the same across all platforms and runs in a web environment.

1.1 The query interface

The query interface provides the means by which users can ask questions of a database in natural language instead of using SQL. This activity is essentially the same, whatever particular query interface is being used. Chapter 9, "Testing the conceptual model" on page 75 shows the query interfaces available with Ergo.

For example, in EPEDEMOE, if users ask such a question as:

Which salesrep has the order with the biggest value?

Ergo will generate this SQL statement for them; they do not need to code it themselves:

```
SELECT X1.ID, X1.NAME, X2.ORDERNO, (X2.QUANTITY * X3.PRODPRICE)
FROM EPE.STAFF X1, EPE.SALES X2, EPE.PRODUCTS X3
WHERE X1.ID = X2.SALESREPNO
AND X2.PRODNO = X3.PRODNUM AND X1.JOB = 'SALES'
AND (X2.QUANTITY * X3.PRODPRICE) =
(SELECT MAX((X4.QUANTITY * X5.PRODPRICE))
FROM EPE.SALES X4, EPE.PRODUCTS X5
WHERE X4.PRODNO = X5.PRODNUM)
```

Ergo users can phrase their questions in different ways that give essentially the same results. Any differences can often be determined from the natural language interpretations.

CUSTOMIZATION TOOL USER'S GUIDE

For example, the first two questions here produce the same natural language interpretation and SQL statement:

What is Quigley's salary?

Find salaries of employees named "quigley".

```
SELECT X1.SALARY,X1.ID,X1.NAME  
FROM EPE.STAFF X1 WHERE X1.NAME = 'OUIGLEY'
```

What is the salary of Quigley?

Find salaries of employees named "quigley".

```
SELECT X1.SALARY,X1.ID,X1.NAME  
FROM EPE.STAFF X1 WHERE X1.NAME = 'OUIGLEY'
```

The next two questions are similar, but the reports that they generate list the columns in a different sequence:

What salary does Quigley have?

Find employees named "quigley" that have salaries.

```
SELECT X1.ID,X1.NAME,X1.SALARY  
FROM EPE.STAFF X1 WHERE X1.NAME ='QUIGLEY'
```

How high salary does Quigley have?

Find employees named "quigley" that have salaries. SELECT

```
X1.ID,X1.NAME,X1.SALARY  
FROM EPE.STAFF X1 WHERE X1.NAME ='QUIGLEY'
```

The following question does not mention Quigley's salary, but the conceptual model states that employees earn commissions, incomes, and salaries. The natural language engine therefore provides three interpretations from which the user can select.

What does Quigley earn?

Find commissions that are earned by employees named "quigley".

```
SELECT X1.COMM,X1.ID,X1.NAME  
FROM EPE.STAFF X1  
WHERE X1.NAME ='QUIGLEY' AND X1.COMM IS NOT NULL ;li.
```

Find incomes that are earned by employees named "quigley".

```
SELECT (X1.SALARY + X1.COMM),X1.ID,X1 .NAME  
FROM EPE.STAFF X1  
WHERE X1.NAME ='QUIGLEY' AND X1.COMM IS NOT NULL
```

CUSTOMIZATION TOOL USER'S GUIDE

Find salaries that are earned by employees named "quigley".

```
SELECT X1.SALARY,X1 .ID,X1.NAME  
FROM EPE.STAFF X1 WHERE X1.NAME ='QUIGLEY'
```

1.2 The natural language engine

The natural language engine is the component that analyzes natural language questions and translates them into SQL statements. When customizing and testing models, it is helpful to have a general understanding of the way that the natural language engine processes questions.

Figure 2 on page 13 shows the main steps that the natural language engine goes through when analyzing questions. The first steps are the parser and semantic analyzer, which take a question and try to determine its semantic structure. It does this from the information about terms and relationships defined in the conceptual model, from its own base vocabulary, and from the rules of grammar for each language held within the engine.

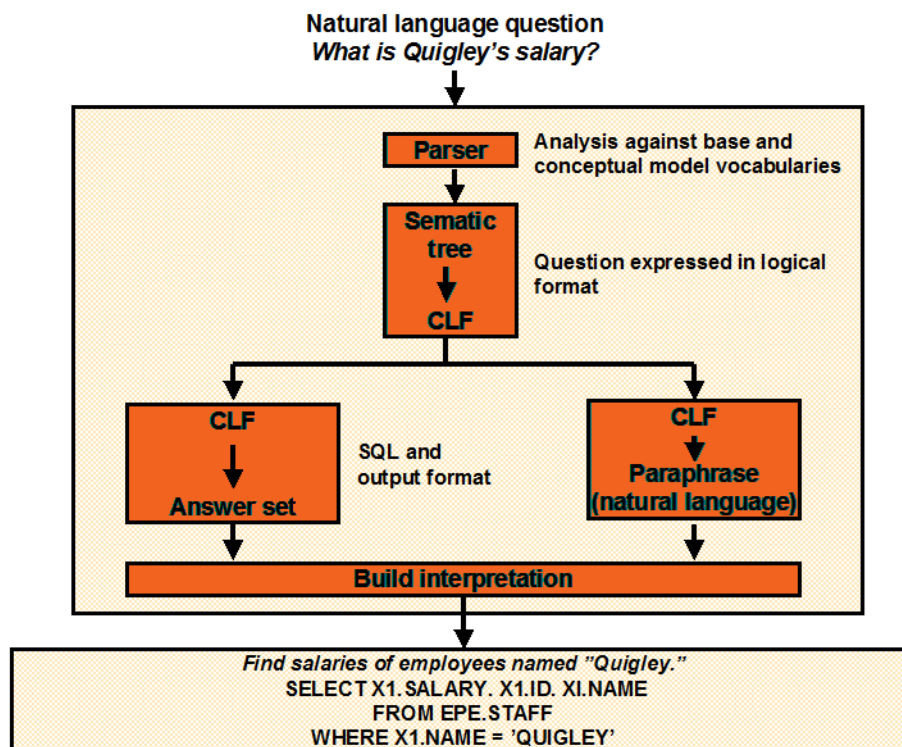


Figure 2. The natural language engine.

If the engine cannot find a semantic connection between all the parts of the question, it produces an error message. But when a valid semantic structure is found, the engine goes on to produce the natural language interpretations and the SQL statements in parallel.

Questions often contain implicit or ambiguous information. So the engine produces a natural language interpretation that endeavors to make explicit what was unspecified in the question. In this way, the user can resolve any possible ambiguities in the question.

CUSTOMIZATION TOOL USER'S GUIDE

When producing the SQL statements for the interpretations, the engine uses information provided in database mode during customization and in the SQL statements defined in language mode. Except for data type checking, this information is not used during the first two steps. The engine first produces basic SQL statements using join paths between the tables, if necessary. It then attempts to optimize the SQL statements using inclusion dependency information, if this is relevant. "Defining dependencies" on page 29 describes how to define these dependencies.

The engine then passes the natural language interpretations and the SQL statements back to the query interface. The user can then select the appropriate interpretation and ask the query interface to send the appropriate SQL statement to the DBMS for execution. The natural language engine is not involved in this process.

1.3 The Customization Tool

The Customization Tool is a graphics editor. You use it to create one or more *conceptual models*.

A conceptual model is a mapping between Ergo and users' natural language questions and the database, as illustrated in Figure 3. The conceptual model contains the words and terms that Ergo users can use in their questions, and a description of how the words relate to each other and to the contents and structure of the database.

Users can have access to several conceptual models, each relating to a particular area. (To query interface users, a conceptual model is called a *language model*.)

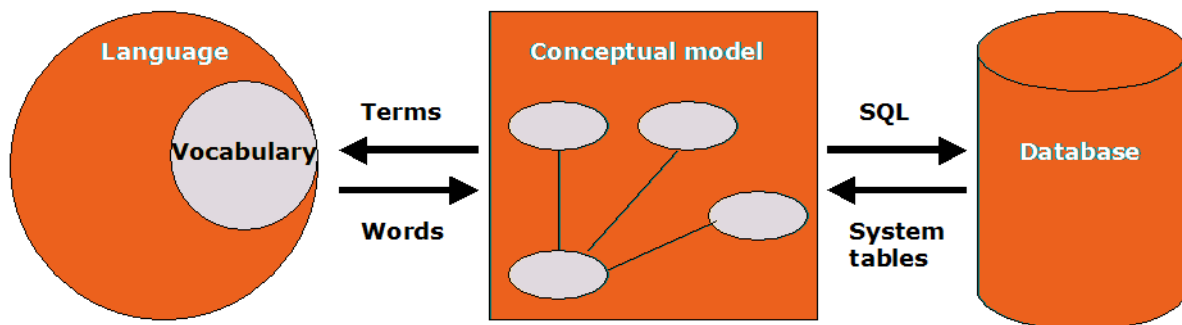


Figure 3. Mapping between database and language.

1.3.1 Basic concepts

The key point to note when working with the Customization Tool is that the conceptual model can be viewed in two ways: from the database and the language perspective. These two views are presented in the Customization Tool as database mode and language mode. Neither of these modes presents information in exactly the same way as an entity-relationship model, but there are some similarities.

CUSTOMIZATION TOOL USER'S GUIDE

In database mode, you define the basic structure of the database, the tables and columns you are using, the keys of the tables, and the dependencies between the tables. Unlike an entity-relationship data model, which generally focuses attention on the tables or entities, the columns or attributes of the tables are shown in database mode with an explicit icon.

When switching from database to language mode for the first time, these tables and columns all become entities, called table and column entities, respectively. This is because from a language perspective there is no difference between the concepts that are mapped to the tables and those mapped to the columns. Concepts that map to tables are called main concepts. Another example of a main concept is a column that has other columns that are functionally dependent on it in a table that is not in third normal form. You can also add other entities to your conceptual model that define other linguistic relationships to the database.

There are two basic types of entity: direct and indirect. Direct entities are those that contain an SQL statement and thus map directly to the database. Indirect entities do not have an SQL statement, and can only be referred to in questions if there is a relationship to another entity that does contain an SQL statement. Examples of indirect entities are table entities, verbs, and composite entities that map to multiple columns.

An entity in language mode can be looked at in a similar way to an entity in an ER model in that all entities have properties or attributes and relationships to other entities. However, unlike ER data models, these properties do not correspond to columns in a database, and the relationships are linguistic rather than dependency connections.

The entity properties are:

Name	The unique name of the entity in the model.
Class or instance	The class of the entity if it represents a general concept, such as a location, quantity, or employee, or an instance of some particular thing: for example, headquarters is an instance of department. There is a basic hierarchy of classes that you add to when you define entities. The class partially determines what type of relationships the entity can enter into.
Category	The category or part of speech of the entity. These are noun, adjective, verb, and proper name.
Term	The word or words that are used in natural language questions. You can define any number of synonyms for terms.
Syntax	Specifies what prepositional relationships can be used with the entity and the case relationships of verbs
SQL statement	Specifies the SQL statement for direct entities.

The types of relationships that entities can enter into are dependent on the class and syntax of the pair of entities involved. The principal relationships, which are described in the text, are:

- Identifier relationship
- Name relationship
- Refer-to relationship

CUSTOMIZATION TOOL USER'S GUIDE

- Associated relationship
- Possessive relationship
- Four locative (place) relationships
- Three temporal (time) relationships
- Duration relationship
- Measured-by relationship
- Counted-by relationship
- Prepositional relationships
- Verb-case relationships

CUSTOMIZATION TOOL USER'S GUIDE

2 PLANNING AND PREPARATION

2.1 Planning a Ergo project

An Ergo model development team must consider these tasks when planning the model:

1. Determining the scope of the project:

- Business factors
- Database factors

2. Pre-customizing:

- Usage assessment
- Language review
- Database study

3. Customizing:

- Processing the tables and columns, and the relationships between them
- Defining subsets of the data. Defining multicolumn entities
- Defining other indirect entities

4. Project evaluation:

- Reviewing the language coverage of the model
- Extending the scope of the model

2.2 Determining the scope of the project

The approach to building an application depends, among other things, on whether the application is database-centric or corpus-centric. A corpus in this context is a set of questions or commands with the corresponding answers or computer commands that define the universe of discourse (UOD).

In the database-centric approach the structure and contents of the database is given from the beginning. You can only ask questions (or issue commands) that are in the database. As an example information about audio files making up your music collection in a smartphone is often held in a relational database. Each record in the database contains information about the name of the song, the album, artist, genre, etc. in the database columns. These columns represent all the things you can ask about e.g. a song. If the information isn't in the database you can't ask about it and thus the database defines your UOD.

c

Figure 4 Sample database for music (audio) files.

In the corpus-centric approach you want to cover an area in e.g. the real world. As an illustration you might want to develop an application which answers questions about how to operate your smartphone. In this case the application must be made aware about concepts like an operating system, applications, an LCD screen, a battery, processing power, memory etc. These concepts describe the UOD, which might be very big. In order to limit the UOD to what is suitable for the users (you wouldn't necessarily want to be able to answer questions nobody is ever going to ask) you want to scope out the domain to cover. The first step is to gather information about what users are asking about (or want to do) in order to build a corpus. Based on the corpus the customizer defines a database structure, including tables and columns. The records in the database are entered during the customization process.

As an illustration to the concept consider the SOIC project (see <http://www.soic.se/en/>), where a group of sailing enthusiasts built a 58 meters long 1:1 replica of a ship Götheborg that was used for trade between China and Sweden during the 18th century. During the construction of the ship visitors were allowed to come and visit the construction site. An automatic support system enabled the visitors to ask question in plain language about the ship and her maiden voyage. Initially a set of questions users might want to ask were identified and further developed through user studies. The questions included things like:

How long is the ship?

CUSTOMIZATION TOOL USER'S GUIDE

What is the weight of the ship?
When will Götheborg cast off?
How is the crew recruited?
etc.

You need not consider different ways of asking the same questions since that is taken care of by the Ergo engine. Based on an analysis what users were asking entities and concepts can be identified and a database structure defined that will allow you to retrieve answers to the users' questions. In the case of Götheborg a database with two tables was defined, containing a url for the answer to each specific question.

From this point and onwards in the process, the two approaches are similar, although not identical, in that they start from a pre-defined database structure.

When approaching customization for the first time, you should define the scope of the model and the criteria for completion. It is advisable to start with a small section of the database as the scope of the initial model

The factors to consider in making a choice are partly business related and partly determined by the characteristics of the database (or corpus) being modeled.

2.3 Business factors

Answer these questions when choosing the model area to be covered:

- What are the objectives of the application, cost savings, quality enhancements, etc.
- Thorough business modeling needs to be done to determine the scope of the application. How much should be covered? How should the application be maintained and updated?
- How will the users use the application and how should the results be presented to the users for best possible effect.

2.4 Database factors

The customizer must thoroughly know the database. Some databases are easier to customize than others, and some database features, especially for database-centric applications, will need more attention than others.

Answer these questions when planning the model:

- How many tables and columns are there?
- In what normal form are the databases?
- Does every table have a primary key?
- How many columns are numerical?
- Do numerical columns have numerical data values?
- How many columns contain coded data?
- How many columns can have NULL values?
- Are dates defined as SQL date data values?
- Does each column have just one meaning?

Dealing with the more tricky aspects of modeling is covered in the chapters on customization.

CUSTOMIZATION TOOL USER'S GUIDE

2.5 Pre-customizing

An important task is to determine through what channels the user will ask his/her questions and how the answers should be presented. An application giving driving directions is probably most useful if the driver can ask for directions with his/her own voice, but the answer should be delivered as a message that the user can read, or even be fed directly into the car's navigation system.

After determining the scope of the model, prepare materials that will help develop the model, test it, and determine when the project goals have been met.

This exercise can be approached from two directions: from the database and from the language that the users will use. Each approach is useful, but together, they provide a solid foundation on which to base your model:

1. Start with an analysis of the tasks that you want the user to perform. You can use this to identify the terms and relationships that will form the basis of your model. From this analysis, you can also create a list of questions that you will need for testing the model.
2. Study your database design. If you have a data model of your database, this can provide much valuable information that will assist you when modeling you can also produce lists of database structures that you can annotate with the terms needed in customization.

2.6 Customizing

After the scope is established, you should follow a systematic approach, for example:

Phase 1: Work with the database; that is, add tables, define column information, and define dependencies.

Phase 2: Define table and column entities and the relationships between them.

Phase 3: Define subsets of data, that is, subclasses, instances, and adjectives.

Phase 4: Define other entities; that is, verbs, composite and calculated entities, and other numerical entities.

Phase 5: Refine the model; for example, add and remove prepositions and add synonyms to primary terms.

These tasks are described in chapters 4 to 8.

2.7 Model testing

To test your model, you must first convert it to the format used by the natural language engine. You can then use one of the query interfaces as outlined in Chapter 9, "Testing the conceptual model on page 75.

It is advisable to test your model both during and after model development. For example, when you have defined the initial tables and columns, you ask simple questions to verify that your model is working as expected, such as:

List employees

List orders

When testing and evaluating, you should ensure that:

- The interpretation makes sense.
- The SQL generated by the questions is correct.
- The SQL executes correctly.

At the later stages of development, you can release the model incrementally to the intended users to

CUSTOMIZATION TOOL USER'S GUIDE

get their comments. Ensure that questions not processed are documented and sent to you for review.

2.8 Maintaining the model

Once the model is in production, you will need to maintain it just like any database or application.

This is especially needed when databases change. Usually, a model must be updated if the database normalization level has changed. But if tables have been added to the database, or if some columns have been added to a table, the effort to update the model is easier, especially if the model is well maintained.

To maintain a model

- Make sure you back up completed work you know is correct.
- Give the latest correct model a recognizably different name from your working models to save time and confusion.
- Document questions that do not work, including any error messages they may generate.
- Document any solutions to difficult tasks you find.
- Do not let too much time pass between Customization Tool sessions, especially while working on an on-going project.
- Make plans for a back-up customizer and make sure pertinent information has been passed on to that person.

CUSTOMIZATION TOOL USER'S GUIDE

3 GETTING STARTED

This chapter describes the steps to follow to start using the Customization Tool. There are three tasks to perform:

1. Installing or launching the Customization Tool component
2. Extracting database catalog information
3. Starting the Customization Tool program

3.1 Installing or launching the Customization Tool component

You install the customization Tool component on your computer by going to dialoguetech.com and clicking on *TOOLS* in the upper right corner. On the tools page which launches click on the Customization Tool item. This installs a Customizer in the browser window on your computer.

NOTE that when you close the Customization Tool it disappears and nothing is stored. You need to store your own work on your computer before you exit the Customizer tool!

3.2 Extracting database catalog information

Before you start using the Customization Tool, you must extract database catalog information about the tables and columns that your conceptual model will use.

Perform this task once for each database, or when the structure of a database changes. For example, if new tables or columns are added to a database, download table information again before updating your conceptual model.

For information on extracting database catalog information, see Chapter 13, "Extracting database catalog information" on page 133 (see also the discussion in chapter 2.2).

3.3 Starting the Customization Tool program

The Customization Tool program launches automatically when you click the Customization Tool icon on dialoguetech.com.

Part 2. Customization tasks

4 WORKING WITH THE DATABASE

The Ergo engine needs information about the database to be able to generate correct SQL queries. This is provided in the form of prolog predicates embedded in the model file you create with the Customization Tool. This chapter describes how you use the database mode to specify the tables and columns that you want to include in the conceptual model. The Customization Tool uses the information you provide to create an initial conceptual model.

There are three ways in the Customization Tool to enter information about the database which will be used in the application:

1. Write prolog predicates defining the database tables and columns in an ASCII text file using an ordinary text editor,
2. Open the Customization Tool and define the database table(s) and columns by writing SQL code directly in the SQL editor window or
3. Extract the database catalog information for the tables to be included in your conceptual model. This is described in Chapter 13, "Extracting database catalog information" on page 133 (see also the discussion in chapter 2.2).

The information you provide about tables and columns is only used in the conceptual model. It does not change the contents or structure of your database.

4.1 Using Prolog predicates

Open an ordinary text editor and write the database definition using Prolog predicates (see <https://sicstus.sics.se/documentation.html> for more information). As an illustration the database used as an illustration in this book is found in Appendix B. The corresponding prolog code looks like:

```
table($EPE$, $ORG$).
table($EPE$, $PRODUCTS$).
table($EPE$, $PROJECT$).
table($EPE$, $SALES$).
table($EPE$, $STAFF$).
table($EPE$, $APPLICANT$).
table($EPE$, $INTERVIEW$).
column($EPE$, $PRODUCTS$, $PRODPRICE$, colinfo($DECIMAL $, $(5,2)$), $n$, $n$).
column($EPE$, $STAFF$, $SALARY$, colinfo($DECIMAL $, $(7,2)$), $n$, $n$).
column($EPE$, $STAFF$, $COMM$, colinfo($DECIMAL $, $(7,2)$), $n$, $n$).
column($EPE$, $INTERVIEW$, $DISP$, colinfo($VARCHAR $, $6$), $u$, $n$).
column($EPE$, $APPLICANT$, $TEMPID$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $INTERVIEW$, $INTDATE$, colinfo($DATE $, $4$), $n$, $n$).
column($EPE$, $INTERVIEW$, $STARTTIME$, colinfo($TIME $, $3$), $n$, $n$).
column($EPE$, $INTERVIEW$, $ENDTIME$, colinfo($TIME $, $3$), $n$, $n$).
column($EPE$, $INTERVIEW$, $MANAGER$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $INTERVIEW$, $FIRSTNAME$, colinfo($VARCHAR $, $9$), $u$, $n$).
column($EPE$, $INTERVIEW$, $LASTNAME$, colinfo($VARCHAR $, $9$), $u$, $n$).
column($EPE$, $APPLICANT$, $COMMENTS$, colinfo($VARCHAR $, $29$), $u$, $n$).
column($EPE$, $APPLICANT$, $EDLEVEL$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $APPLICANT$, $ADDRESS$, colinfo($VARCHAR $, $17$), $u$, $n$).
column($EPE$, $APPLICANT$, $NAME$, colinfo($VARCHAR $, $9$), $u$, $n$).
column($EPE$, $INTERVIEW$, $TEMPID$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $ORG$, $DEPTNUMB$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $ORG$, $DEPTNAME$, colinfo($VARCHAR $, $14$), $u$, $n$).
column($EPE$, $ORG$, $MANAGER$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $ORG$, $DIVISION$, colinfo($VARCHAR $, $10$), $u$, $n$).
column($EPE$, $ORG$, $LOCATION$, colinfo($VARCHAR $, $13$), $u$, $n$).
```

CUSTOMIZATION TOOL USER'S GUIDE

```
column($EPE$, $PRODUCTS$, $PRODNUM$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $PRODUCTS$, $PRODNAME$, colinfo($VARCHAR $, $11$), $u$, $n$).
column($EPE$, $PRODUCTS$, $PRODGRP$, colinfo($VARCHAR $, $10$), $u$, $n$).
column($EPE$, $PROJECT$, $PROJNO$, colinfo($CHAR $, $4$), $u$, $n$).
column($EPE$, $PROJECT$, $ENDD$, colinfo($DATE $, $4$), $n$, $n$).
column($EPE$, $PROJECT$, $STARTD$, colinfo($DATE $, $4$), $n$, $n$).
column($EPE$, $SALES$, $ORDERNO$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $SALES$, $SALESREPNO$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $SALES$, $PRODNO$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $SALES$, $CUSTNO$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $SALES$, $QUANTITY$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $STAFF$, $YEARS$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $STAFF$, $ID$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $STAFF$, $DEPT$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $STAFF$, $JOB$, colinfo($CHAR $, $5$), $u$, $n$).
column($EPE$, $STAFF$, $NAME$, colinfo($VARCHAR $, $9$), $u$, $n$).
column($EPE$, $PROJECT$, $DEPT$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $PROJECT$, $PRODNUM$, colinfo($SMALLINT$, $2$), $n$, $n$).
column($EPE$, $PROJECT$, $TIMESTAMP$, colinfo($TIMESTMP$, $10$), $n$, $n$).
key($EPE$, $SALES$, [$ORDERNO$]).
key($EPE$, $STAFF$, [$ID$]).
key($EPE$, $PROJECT$, [$PROJNO$]).
key($EPE$, $PRODUCTS$, [$PRODNUM$]).
key($EPE$, $ORG$, [$DEPTNUMB$]).
key($EPE$, $INTERVIEW$, [$TEMPID$]).
key($EPE$, $APPLICANT$, [$TEMPID$]).
inc_dep($EPE$, $SALES$, [$SALESREPNO$], $EPE$, $STAFF$, [$ID$]).
inc_dep($EPE$, $ORG$, [$MANAGER$], $EPE$, $STAFF$, [$ID$]).
inc_dep($EPE$, $SALES$, [$PRODNO$], $EPE$, $PRODUCTS$, [$PRODNUM$]).
inc_dep($EPE$, $STAFF$, [$DEPT$], $EPE$, $ORG$, [$DEPTNUMB$]).
inc_dep($EPE$, $PROJECT$, [$DEPT$], $EPE$, $ORG$, [$DEPTNUMB$]).
inc_dep($EPE$, $INTERVIEW$, [$MANAGER$], $EPE$, $ORG$, [$MANAGER$]).
inc_dep($EPE$, $INTERVIEW$, [$TEMPID$], $EPE$, $APPLICANT$, [$TEMPID$]).
inc_dep($EPE$, $PROJECT$, [$PRODNUM$], $EPE$, $PRODUCTS$, [$PRODNUM$]).
join_path($EPE$, $PROJECT$, [$PRODNUM$], $EPE$, $PRODUCTS$, [$PRODNUM$]).
join_path($EPE$, $PROJECT$, [$DEPT$], $EPE$, $ORG$, [$DEPTNUMB$]).
join_path($EPE$, $PRODUCTS$, [$PRODNUM$], $EPE$, $SALES$, [$PRODNO$]).
join_path($EPE$, $INTERVIEW$, [$TEMPID$], $EPE$, $APPLICANT$, [$TEMPID$]).
join_path($EPE$, $INTERVIEW$, [$MANAGER$], $EPE$, $ORG$, [$MANAGER$]).
join_path($EPE$, $STAFF$, [$DEPT$], $EPE$, $ORG$, [$DEPTNUMB$]).
join_path($EPE$, $SALES$, [$SALESREPNO$], $EPE$, $STAFF$, [$ID$]).
```

The first few lines beginning with the predicate table define the table(s) in the database. Save this file to e.g. you hard drive. When you launch the Customization Tool select open file and choose the file where you stored the database definition in the form of Prolog predicates.

4.2 Using SQL to define the database

To be completed.

4.3 Adding tables to the conceptual model

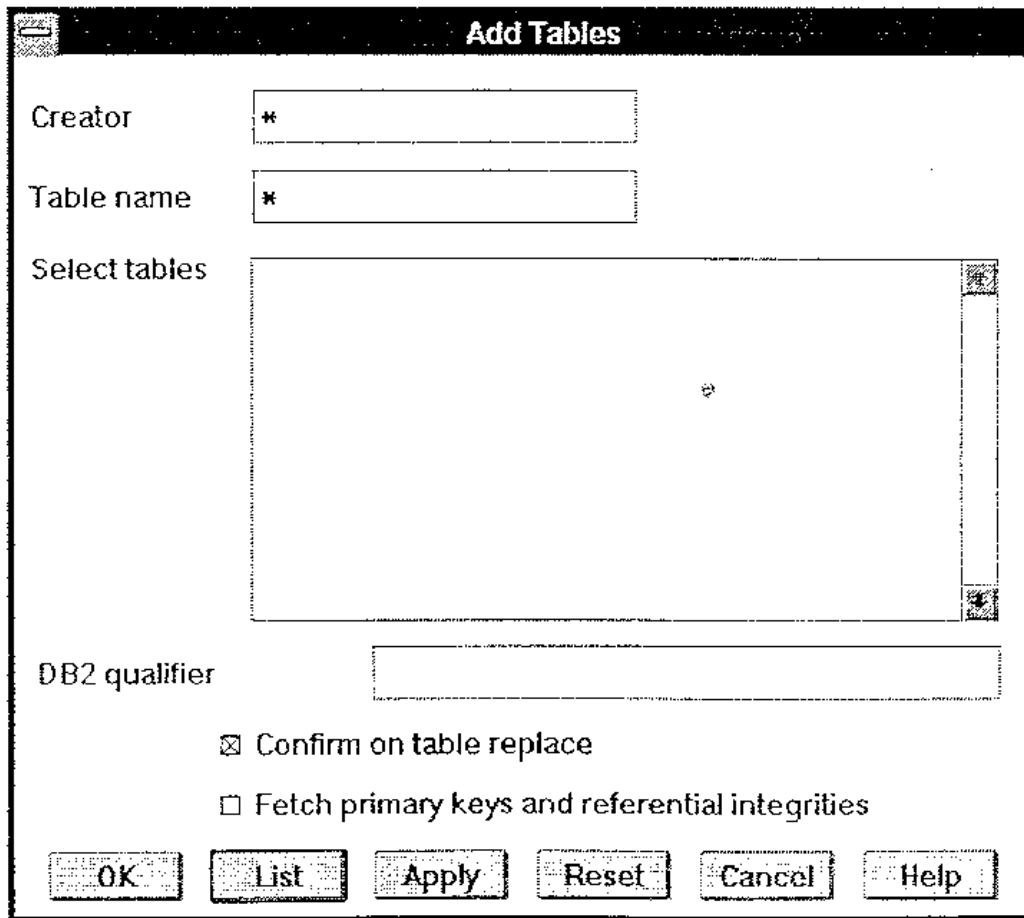
When you extract database catalog information (see Chapter 13) the information includes the names of all tables in your database, their columns, and their data types. Your first task is to select the tables that you need, and add them to your conceptual model.

Unless you have a large number of tables with little connection between them, add all the tables you want at the same time.

Follow these steps to add tables from the database:

CUSTOMIZATION TOOL USER'S GUIDE

1. Select *Database* from *Mode*. If you are adding tables to an existing model, select *Add tables* from *Action*. If you are creating a new model, the *Add tables* window appears automatically:



The screenshot shows a dialog box titled "Add Tables". It has a close button in the top-left corner. The dialog contains the following elements:

- Creator:** A text input field with an asterisk (*) next to it.
- Table name:** A text input field with an asterisk (*) next to it.
- Select tables:** A large empty rectangular area, likely a list box for selecting tables.
- DB2 qualifier:** A text input field.
- Options:** Two checkboxes:
 - Confirm on table replace
 - Fetch primary keys and referential integrities
- Buttons:** A row of six buttons: OK, List, Apply, Reset, Cancel, and Help.

Figure 5. Add tables window.

2. Specify search criteria for the tables that you want to include in the conceptual model by filling in these fields:

Creator To specify the SQL user ID of the table creator
Table name To specify the names of the tables that you are adding.

In all fields, you can use * and ? (global search characters):

* represents any character string (0 or more characters),

? represents anyone character.

3. If you are adding tables to an existing model, select whether you want to automatically update existing tables with the same name, or whether you want to be prompted when duplicate names occur.
4. Select whether you want to automatically define keys and inclusion dependencies from the information in the database catalog tables. To define keys and inclusion dependencies for each column in turn, see "Defining column information" on page 27.
5. Press *List*. The Customization Tool lists all matching tables in alphabetical order.

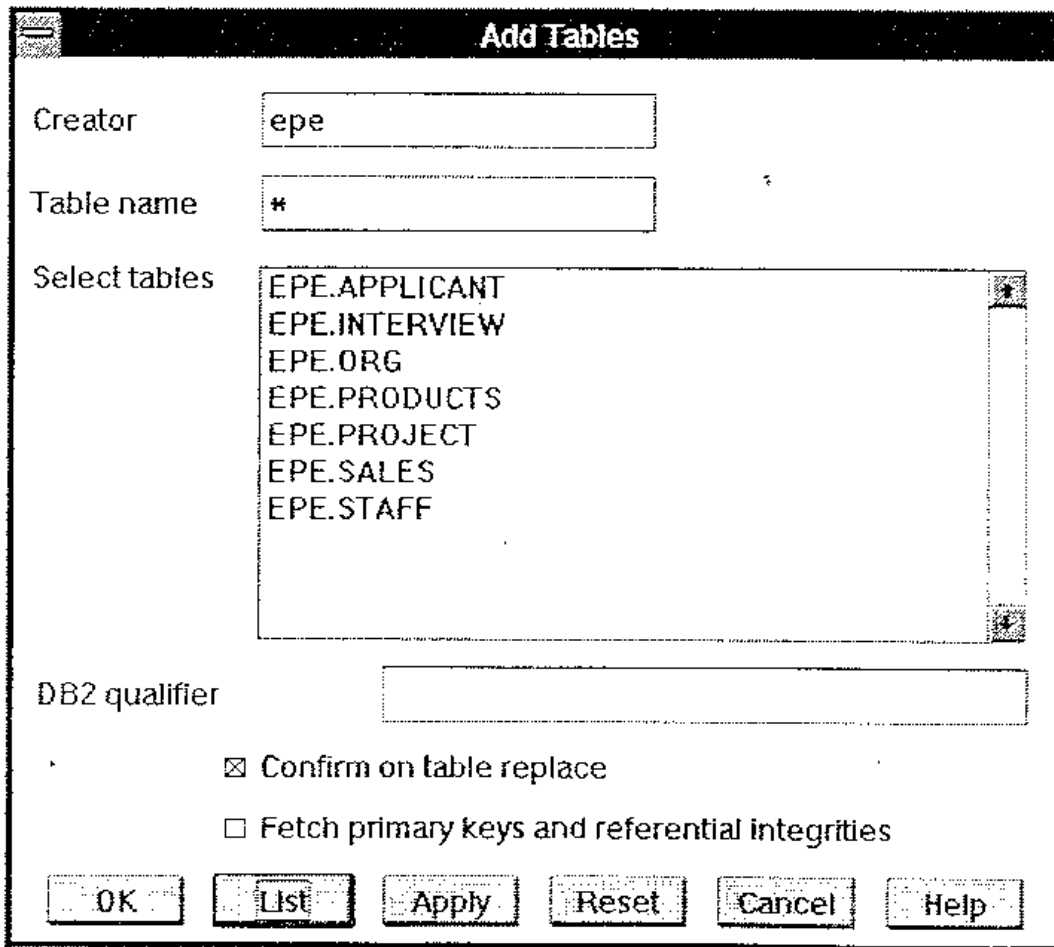


Figure 6. Selecting tables from the Ergo sample database.

6. Click once with the mouse on each table that you want from the list. You can select any number of tables.
7. Press *Apply* to select the tables and keep the window so that you can specify new search criteria and add more tables. Or, press *OK* to select the tables and remove the window.

4.4 Working with the database diagram

Before defining the concepts in the model in language mode, specify a number of properties of the tables and columns in the database in database mode. These include:

- Keys
- Case information
- Join paths
- Inclusion dependencies

The resulting prolog predicates look like the example in chapter 4.1. When you select the tables you want and press *OK* in the *Add tables* window, the database diagram appears in the primary window. It contains an icon for each table that you have added, and for each column

CUSTOMIZATION TOOL USER'S GUIDE

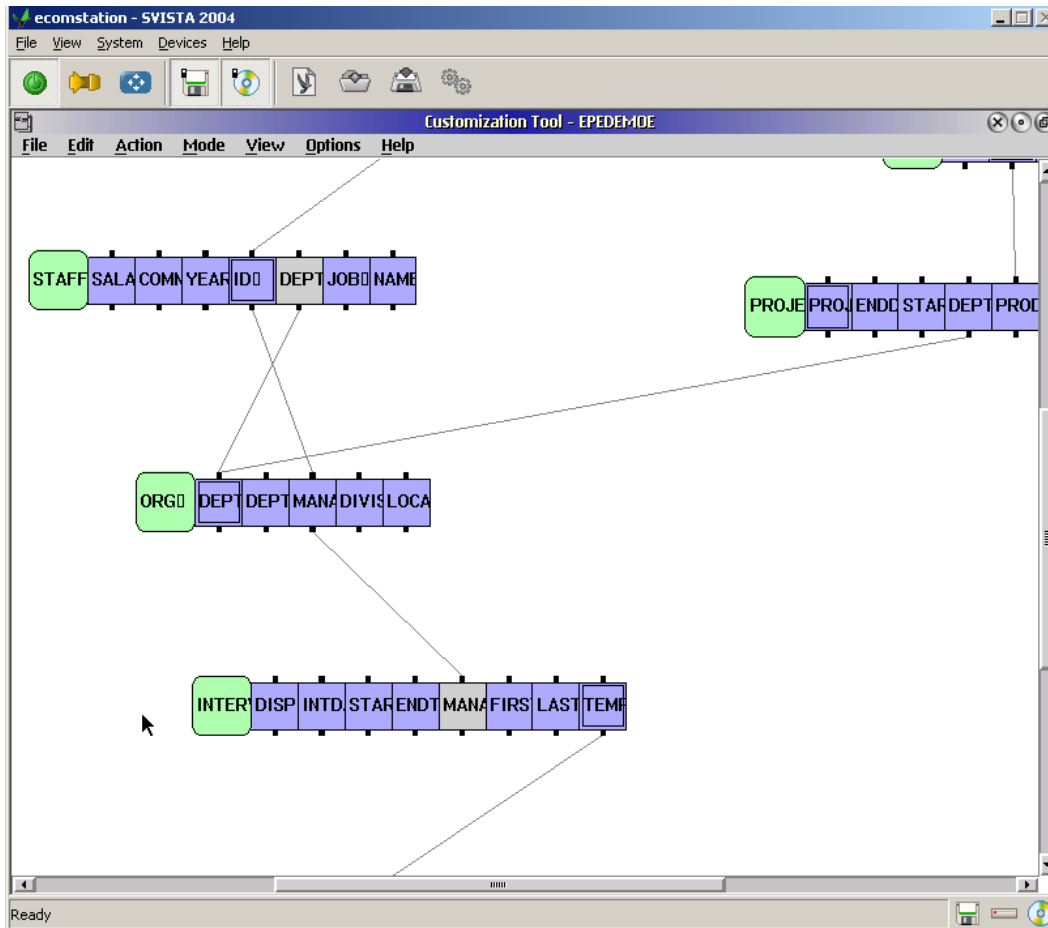


Figure 7. Diagram with table and column icons.

Use the mouse to specify column information and dependencies:

- Select a table or column icon by clicking on it once with the left mouse button.
- Define a column by double clicking its icon with the left mouse button.
- Move a table by holding down the right mouse button and dragging the table icon.
- Expand a collapsed table by double clicking on it with the left mouse button.
- Create a dependency between two columns by holding down the right mouse button and dragging one column icon onto the other column icon.

4.5 Hiding columns

If the tables in your database contain columns that you will not use when defining the conceptual model, you can hide the columns that you do not need. To do this:

1. Select the table icon of the table whose columns you want to hide.
2. Choose *Hide columns* from *Action*. The *Hide Columns* window appears:

CUSTOMIZATION TOOL USER'S GUIDE

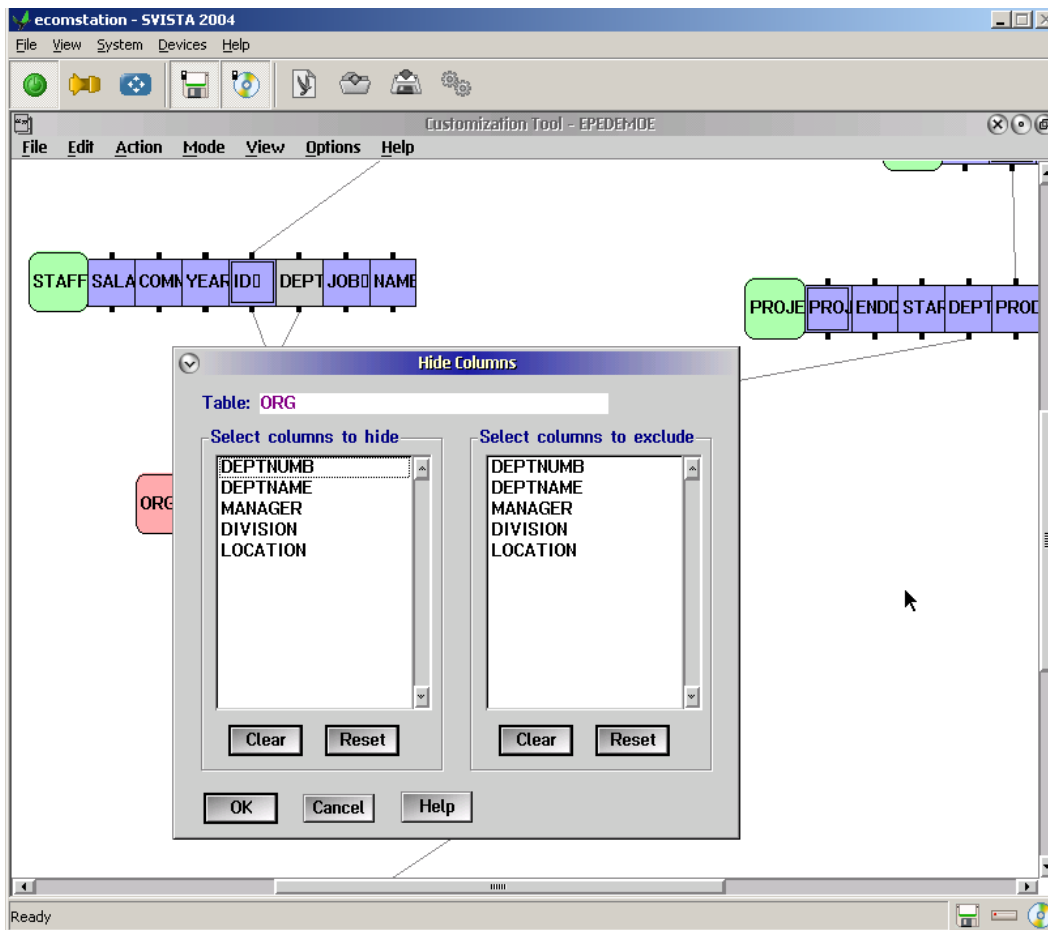


Figure 8. Hide columns window.

3. Select the columns that you want to hide from the list on the left. These columns will not appear in either database or language mode. Alternatively, you can exclude columns by using the list on the right. These will be shown in database mode, but not in language mode. See "Excluding columns" on page 29 for advice on when to exclude columns.

4. Press *OK* to accept the actions.

Use *Clear* to show all columns or *Reset* to undo the settings you have made since displaying this window. Use *Cancel* to remove the window without performing any action.

Note: Ergo does not support long string columns. These are columns defined with a data type VARCHAR and a length greater than 254 or a data type VARCHAR and a length greater than 127. Columns with these data types are automatically hidden and are not available in the model.

4.6 Defining column information

Use the *Define Column* window to specify column information. It appears when you double click on a column icon or select *Define* from *Action* when a column icon is selected.

CUSTOMIZATION TOOL USER'S GUIDE

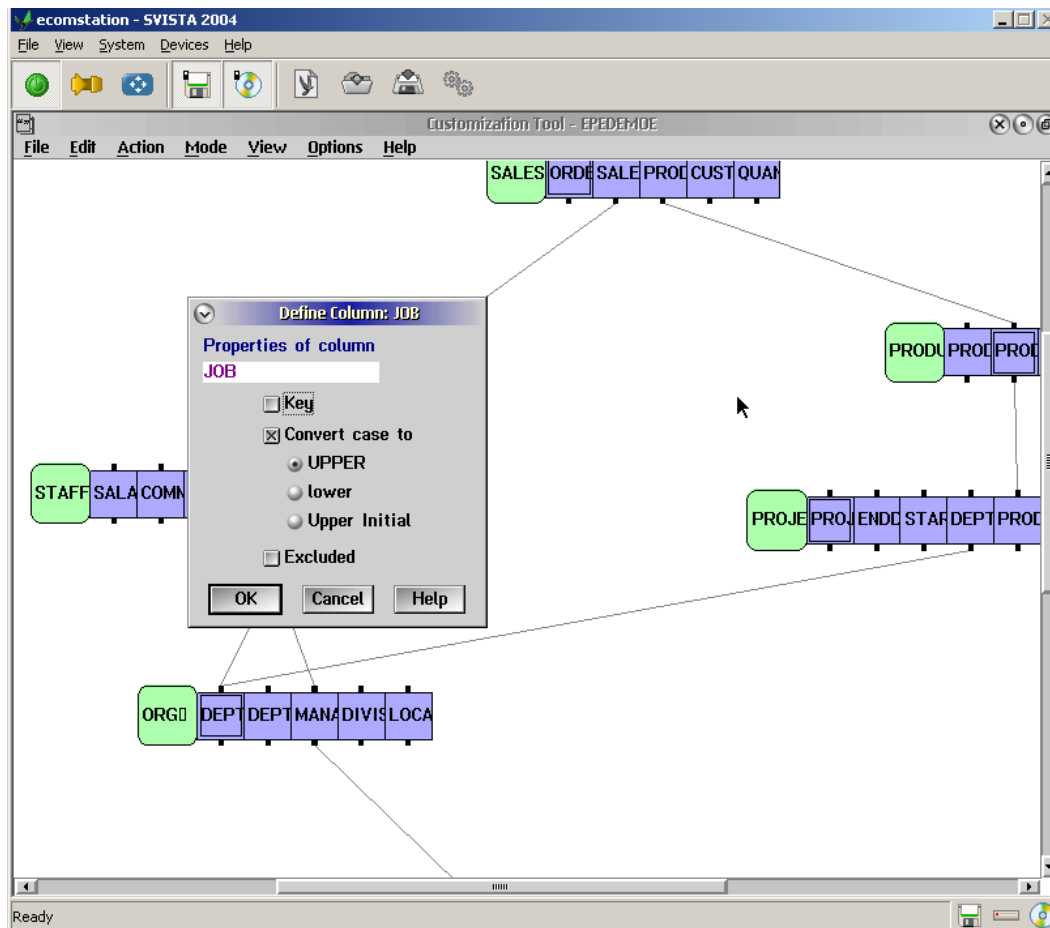


Figure 9. Define column window.

4.7 Defining keys

The primary key of a table is a one or more column that uniquely identifies the rows in the table. You should define the primary key of each table in the model if this was not done automatically when you added tables. This is necessary when the DBMS that you are using does not explicitly support keys. If you define more than one column as a key, Ergo will assume that the table has a multicolumn key.

When you select *Key*, the Customization Tool adds a second frame to the column icon to indicate that it is a key.

4.8 Defining case information

If the column has a CHAR or VARCHAR data type you can specify the case in which the data in the column is stored. The engine will then convert the case of the data values in users' questions to match the case in the columns in the database. The default settings are conversion to upper case. If you turn off automatic conversion, the user must enter data values exactly as they are in the database.

Note that any columns linked through join paths or inclusion dependencies must have compatible case information.

If you select automatic case conversion, these are the options that you can select:

UPPER for columns where the data values have uppercase characters, such as LOS ANGELES.

lower for columns where the data values have lowercase characters, such as los angeles.

CUSTOMIZATION TOOL USER'S GUIDE

Upper Initial for columns where the data values have an initial uppercase (first letter of every word), such as Los Angeles.

4.9 Excluding columns

You can exclude columns that are needed in database mode, but are not required in language mode. A column that you exclude remains in the database diagram, but its style changes to remind you that it is excluded. If you want to remove columns from both the database and language diagrams, you use hide columns described in "Hiding columns" on page 26.

You should exclude columns that are linked to another in a join path when the concepts that they represent are identical. If, however, a column is a subset of another and this subset characterizes another concept with its own relationships, you should not exclude the column.

For example, in EPEDEMOE, the column DEPT has been excluded from the STAFF table because this denotes exactly the same set of values as DEPTNUMB in the ORG table. However, MANAGER has not been excluded from the ORG table because this is a concept that is a subset of (employee) ID in the STAFF table and is used in the model with its own relationships, such as:

managers lead projects

managers interview applicants

You should note that different concepts are not necessarily denoted by individual words. For example, TEMPID in the INTERVIEW table has not been excluded from EPEDEMOE because interviewed applicants are logically a subset of all applicants. Here the adjective interviewed is used to qualify the concept of applicant.

Note here the difference between the logical relationship between tables and their contents at any one time. In the EPE database tables, see Appendix B, all applicants are interviewed. But this does not mean that applicants and interviewed applicants are logically equivalent.

4.10 Defining dependencies

There are two types of dependency relationship between or within tables that you can define. They are join paths between tables and inclusion dependencies both between and within tables. These dependencies are used by the natural language engine when generating SQL statements after users' questions have been interpreted.

If you choose to add referential integrity information in the Add tables window, described in "Adding tables to the conceptual model" on page 10, the Customization Tool automatically defines inclusion dependencies between all foreign keys and their respective primary keys. Join paths are not added automatically.

Otherwise, you create a dependency by dragging one column icon onto another column icon, using the right mouse button. When you release the mouse button, the Define dependency window appears:

CUSTOMIZATION TOOL USER'S GUIDE

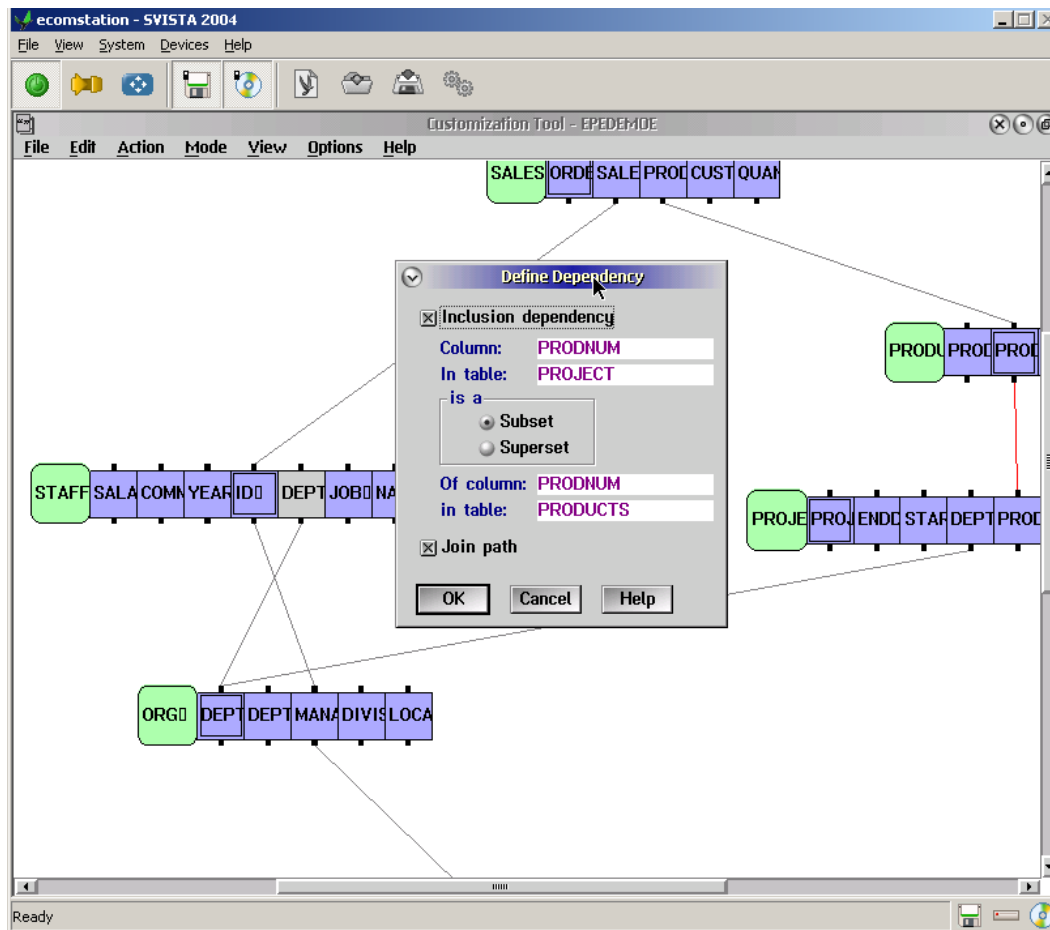


Figure 10. Defining an inclusion dependency.

After you define a dependency, the Customization Tool will draw a line between the two columns. This is done automatically when you choose to add referential integrity information in the *Add tables* window.

You can only select a dependency between tables whose columns are displayed. If a table is collapsed, the dependency lines become selectable when you expand the table.

To modify a dependency, you double click on the dependency line or you select it and then select *Define* from *Action*. Alternatively, you can use the drag and drop technique used to create the dependency initially

If you want to delete a dependency between two columns, you double click on the dependency icon, and deselect both *Inclusion dependency* and *join path*.

You can only define dependencies between columns with compatible data types. These are:

- Same data type
- SMALLINT, INTEGER, DECIMAL, and FLOAT
- CHAR, VARCHAR, and DATE
- CHAR, VARCHAR, and TIME
- CHAR, VARCHAR, and TIMESTAMP

The remainder of this section provides guidelines about specifying join paths and inclusion dependencies. You should define both of these when they are both applicable.

CUSTOMIZATION TOOL USER'S GUIDE

4.11 Defining join paths

If you define linguistic relationships, such as possessive and locative relationships, between columns in different tables in language mode, you must define join paths. You can define join paths either directly between the two tables involved in the linguistic relationship or indirectly. In this latter case, you must define join paths between the two tables and another table. That is, if you want to linguistically relate tables A and B, you can indirectly join them by creating join paths between tables A and C and tables B and C.

However, if you define one column as a subclass of another, and there are no other relationships between the columns in the tables, there is no need to specify a join path.

Join paths are normally links in the database between primary and foreign keys, but not necessarily so. Foreign key columns are columns containing data values that are found among the data values of a primary key in another table.

For example, in EPEDEMOE, column SALESREPNO in table SALES is a foreign key corresponding to the primary key ID in table STAFF. A join path has been defined between the two columns because linguistic relationships, such as sell and buy, have been defined between entities representing columns in the two tables.

If the table contains a multicolumn key that is part or all of a foreign key, then connect each pair of matching columns with a join path. When there is more than one join path between tables, Ergo interprets this as a join path between multicolumn keys.

If you have excluded a column in a table because it has a foreign key, you must define a join path between the two columns. Otherwise, users will not be able to obtain information from both tables.

If there are potentially two join-paths between two tables, you should choose the one that joins the most general concepts. For example, in EPEDEMOE, the MANAGER column in the ORG table is a foreign key for ID in STAFF, and the DEPT column in STAFF is a foreign key for DEPTNUMN in ORG. Here you define a join-path between the department-columns. If you define a join path between MANAGER and ID, users will get information relating to managers only and not to all employees, when asking questions about employees in departments. The results are unpredictable if you define both possible join paths.

When you define a join path, you should usually also define an inclusion dependency. However, if one column is not a foreign key of the other, do not define an inclusion dependency. This situation can arise when two columns are intersecting subsets of a third set that does not appear in the model. For example, if you have tables with columns representing managers and sales reps, but not employees, then you may need to define a join path between managers and sales reps. However, it would be a mistake to define an inclusion dependency between the two columns.

4.12 Defining inclusion dependencies

While join paths provide essential links between tables to let Ergo retrieve information from two or more tables, inclusion dependencies are optional. The purpose of inclusion dependencies is to simplify the SQL statements generated from users' questions, and so improve the performance of data retrieval.

You define an inclusion dependency between two columns when one column is a subset of the other or if the columns contain exactly the same values. This is always the case when one column is a

CUSTOMIZATION TOOL USER'S GUIDE

foreign key of the other. It is never wrong to define inclusion dependencies in these cases. The Customization Tool automatically defines inclusion dependencies between such columns if foreign keys are defined in the database and if you select to fetch referential integrity information when adding tables, described in "Adding tables to the conceptual model" on page 23.

To define an inclusion dependency, select *Inclusion Dependency* in the *Define Dependency* window (see Figure 10 on page 30). Then select subset or superset to specify the direction of the inclusion dependency.

EPEDEMOE includes several examples of inclusion dependencies, such as: PRODNUM in PROJECT as a subset of PRODNUM in PRODUCTS and TEMPID in INTERVIEW as a subset of TEMPID in APPLICANT. To see them all, click on any of the connecting lines in database mode with model EPEDEMOE.

In general, you define inclusion dependencies between two different tables. However, you can also define intra-table inclusion dependencies. This could occur when one column is a subset of another in the same table.

This situation may well occur in hierarchically structured tables, such as those showing the relationships between organizational units in a company and in bill-of-materials applications, or in databases that are not in third normal form.

For example, in the table in Figure 11, the data values in SUPERIOR are a subset of the data values in DEPT. This superior department column can therefore be defined as an inclusion dependency of the department column. Defining the inclusion dependency here can simplify the generated SQL.

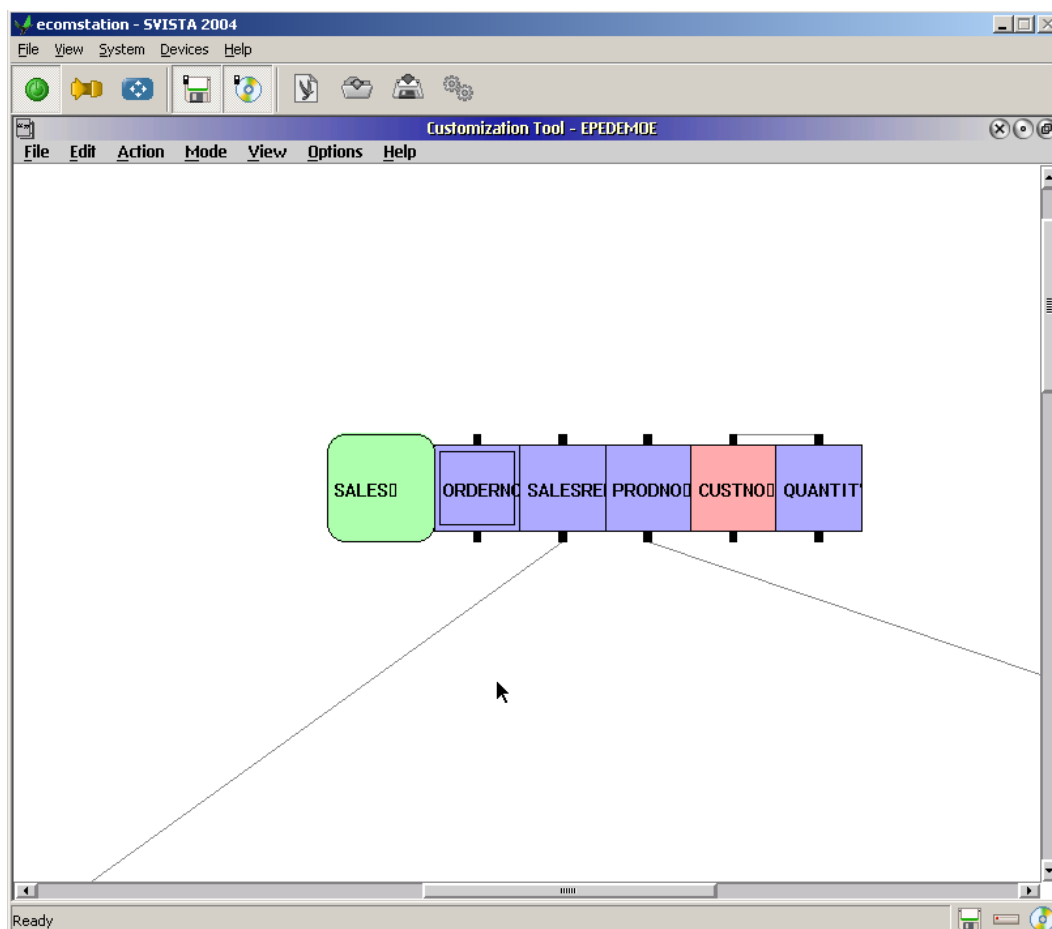


Figure 11. Intra-table inclusion dependency.

CUSTOMIZATION TOOL USER'S GUIDE

The diagram in Figure 12 on page 33 shows some of the EPE sample tables with inclusion dependencies, join-paths, and excluded columns:

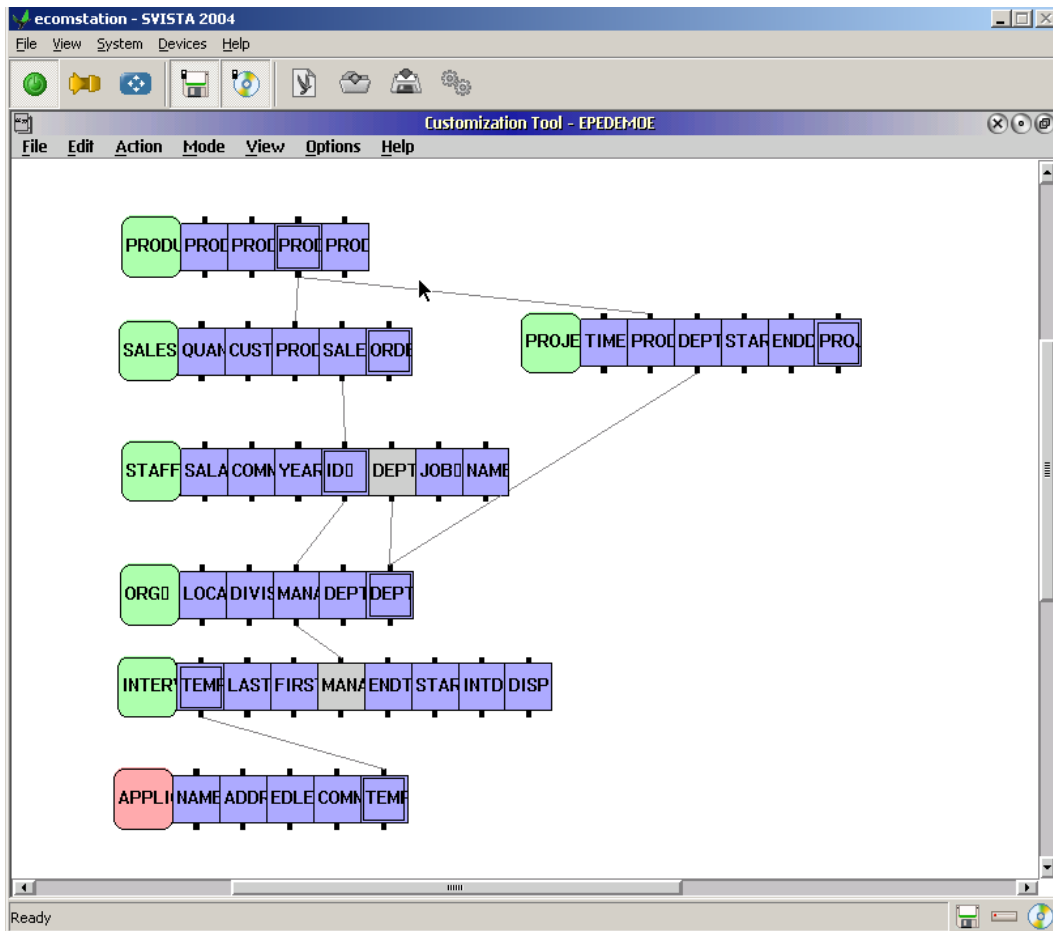


Figure 12. Table diagram with dependencies, keys, and excluded columns.

4.13 Moving between database mode and language mode

When you have added tables, and defined column information and dependencies, go from database mode to language mode to create the conceptual model, as described in the following four chapters.

When working on a conceptual model in language mode, you can return to database mode to:

- Extend the conceptual model by adding more tables to it.
- Update the model when the structure of the database has changed. For example, if a column is added to one of the tables in the model, replace the previous version of the table with the new version.

Before updating the model, extract database catalog information from the database once again. (See Chapter 13, "Extracting database catalog information" on page 133.)

Changes that you make in database mode do not affect the existing entities in a conceptual model. For example, if you go from language mode to database mode and exclude a column, the column entity remains in the conceptual model.

CUSTOMIZATION TOOL USER'S GUIDE

5 DEFINING TABLE AND COLUMN ENTITIES

Having loaded the tables and columns that you need in your model, and processed them in database mode, you are ready to begin the main part of the customization process - defining the conceptual model itself. The most natural place to start with this activity is to define the entities that relate directly to the tables and columns in the database. This chapter describes this process, which has two main steps:

1. Naming and identifying table entities
2. Defining the other concepts relating to the tables and columns and the relationships between them

In a database many of these relationships are implicit because a table consists of a number of related columns or attributes of the entity. However, to let the user ask questions in natural language, these relationships must be made explicit.

During this activity you do not need to define the SQL statements that relate to these tables and columns. Ergo automatically provides SQL statements for all columns, and SQL statements for table entities are generally unnecessary. Modifications to the SQL statements for the columns that you might need can be made later when new entities requiring their own SQL statements are created.

However, think about the way the concepts will be used in the SQL statements generated by the natural language engine while customizing at this phase. In particular, notice when you are defining a concept:

- How it will be used in the SQL SELECT clause to specify the columns to be displayed.
- What role the concept has in the WHERE clause, for example, which defines the row selection criteria.

5.1 Defining, naming, and identifying table entities

To begin the main part of the conceptual modeling process switch to language mode. The first task is to define, name, and identify the table entities.

You can best start with those tables that represent the main concepts in your model, such as employees, customers, and suppliers. In general, these tables will have both an identifier, such as an employee ID, and a name, such as an employee name. Ergo directly supports tables with such a structure. However, some tables have different structures, when you have further options about how to customize the table.

You should make a distinction between the entity names that are used internally within the Customization Tool to uniquely identify an entity, and the column entity that names a table entity, such as employee name. Process every table in the conceptual model in this way before going on to deal with the other columns in the tables.

If the tables in your database follow the rules of the relational model of data, then one or more columns in the table will be a primary key that uniquely identifies the rows in the table. Define the primary key as the identifier of the table. An employee name column is a typical example of a table name, if there is one.

The tasks that you perform to define the names and identifiers of tables include many basic Customization Tool operations. Once you have completed this part, you will better understand how the Customization Tool functions as a whole.

These are the main tasks:

CUSTOMIZATION TOOL USER'S GUIDE

1. State the internal name of the table and column entities.
2. Classify the table and column entities.
3. Specify terms for the table and column entities.
4. Define the relationships between the table entity and the column entities.

In practice, it is best to perform steps 1-3 for the table entities first and then to repeat these for the column entities. You can then perform step 4 to relate the table entities to their corresponding column entities.

5.2 Naming entities

The names of the entities or concepts in the model help identify the entities within the Customization Tool. For concepts that are directly based on the tables and columns, Ergo provides a name based on the table or column name.

If there are two columns with the same name in different tables, then the Customization Tool adds a number at the end of each name to distinguish one name from the other. The names must be unique within the model as a whole, not just within one table.

Note: The names do not affect the kind of questions that you can ask on the query interface.

Naming conventions can help identify entities, for example, if the entity name is similar to the primary term associated with the entity. You can also add an extension to identify how the entity relates to the database; for example, you could use `_col` for column entities, `_table` for table entities, and so on. Or you can devise your own naming conventions.

5.3 Direct and indirect table entities

When a table has both an identifier and a name, you customize the table entity as the main concept. This is an example of an indirect entity when there is no direct relationship between the entity and a column in the database. The meaning of the entity is derived from the relationships that it has to, other entities that have a direct relationship to columns in the database. Direct entities have an SQL statement associated with them, while indirect entities do not.

If you have a table that has only a name or an identifier, then you can ignore the table entity altogether, and define the main concept as the name or identifying column. The main concept is then a direct entity with a direct mapping to the database.

For example, in a countries table, all the country names would be unique and therefore also serve as identifiers. In such a table, you can define the main concept country as either the table entity or the column entity that contains the name of the country.

5.4 Classifying names and identifiers

When you classify an entity, you associate that entity with a concept in the classification hierarchy.

In general, the class that you give to the table affects many other relationships than just the name and identity of the table. Unless you want to jump forward to "Classifying entities" on page 38, you can temporarily classify the table entity as the most general type: thing.

The concepts that denote the name and identity of the table are simply classified as name and identifier, respectively.

Sometimes the identifying column also serves as the name of an entity. For example, country in a country table can be both an identifier and a name because one and only one country exists in this world having one and only one name. In these cases, do not define the column as both a name and

CUSTOMIZATION TOOL USER'S GUIDE

an identifier unless you want to use name terms, as described in "Defining terms for names and identifiers", just classify it as identifier.

If the primary key of the table covers multiple columns, create a composite entity to identify the table. (See "Composite identifiers" on page 62.) Similarly, if the name spans more than one column, as it would if first and last names of employees were in separate columns, define a composite entity as the name of the table. The 3 individual columns of the composite identifier or name need not be classified as identifiers or names, respectively. Classifying them as things is sufficient.

The names and identifiers of tables are not necessarily the only examples of these types of concept in the model. If a table is in first normal form (1 NF) or 2NF, but not in 3NF, then some column may be the name or identifier of another column in the table. If you have not defined a view to handle this situation, you must define a new main concept to represent the implicit table entity in the table.

5.5 Defining terms for names and identifiers

Although, in general, you must define a term and its part of speech before you can define relationships for an entity, this is not necessary in the case of names and identifiers. But you must define a term for the main concept if you use it in a question.

The terms *id* and *identifier*, are automatically available for identifiers as part of the base vocabulary, should you need them. If you want to use these terms in relationship to the table entity, you must define a possessive relationship with that entity. The terms *name*, *first name*, and *last name* are also automatically available as part of the base vocabulary for names. In this case, you do not need to define any additional relationships.

If you do want to give terms to table entities and their names and identifiers, you can do so. You can then define any relationships to these entities, as is described in "Processing the other column entities" on page 38.

5.6 Defining name and identifier relationships

Before you can use names and identifiers in questions, you need to define their relationships to the table entity.

To define an identifier relationship, you select the relationship:

What uniquely identifies <table entity>? <column entity>

To define a name relationship, select the relationship:

What is the name of <table entity>? <column entity>

If you select the name relationship, the words described in "Possessive words" on page 44 are automatically available. Do not also select the possessive relationship. If you select both the name relationship and the possessive relationship, users may receive duplicate interpretations for their questions.

When Ergo answers a question about an entity, the answer contains both the identifier of the entity (if one exists) and the name of the entity (if one exists).

Example: In EPEDEMOE, staff name has this relationship:

What is the name of staff? staff name

These tasks are described in more detail in "Defining relationships" on page 43.

You can now upload the model and test this initial stage of customization by asking such questions as:

CUSTOMIZATION TOOL USER'S GUIDE

List employees
List orders

You should get a list of the identifiers and names of each row in the table. This is because the names and identifiers of tables act like components of a composite table entity, if both are defined, on output. To output other columns with a table entity, you can specify these columns using the associated relationship described in "Associated relationships" on page 64.

When used as search criteria, the names and identifiers of the table are implicitly defined from the table.

For example, in EPEDEMOE, these questions:

Who is manager of [department] north?, and
Who is manager of [department] 84?

give these natural language interpretations, among others:

Find employees (that are department managers) of departments named "north".
Find employees (that are department managers) of departments with identifiers that are 84.

Ergo has implicitly recognized that north and 84 are the name and identifier of department, respectively, because one is numeric and the other is not. If both the name and identifier are nonnumeric, the questioner can explicitly state that north is the name of the department by asking:

Who is the manager of the department named north?

because named is a term in the base vocabulary.

5.7 Defining refer-to relationships

Alternatively, when there is an ambiguity between names and identifiers of table entities, you can resolve the ambiguity by using the refer-to relationship.

For example, if you have a table of courses in which both course name and identifier are characters, you can specify that the identifier column, rather than the name column, should be used when asking questions about courses. You do this by selecting this relationship:

What does course-id refer to? course

This relationship allows this question:

Who teaches course CEB107?

which gives a natural language interpretation such as:

Find teachers that teach courses with the identifier "CEB107".

If you also want to ask questions referring to course names, you can enter questions like these:

Who teaches courses named Spanish?
Who teaches Spanish courses?

Alternatively, you can also define a refer-to relationship between the course name and the table entity. However, you would then get two interpretations for the question:

Who teaches course CEB107?

CUSTOMIZATION TOOL USER'S GUIDE

5.8 Processing the other column entities

Once the initial basic task is finished, define concepts for all the other columns in the tables in your model and the relationships between them. This last step is crucial, because the questions that you can ask will be severely constrained until the relationships are defined.

With one optional addition, the main activities are the same as those described in "Defining, naming, and identifying table entities" on page 34. Specifying the syntax, or prepositions, for the nouns involved in the relationships may also be necessary. However, many of these prepositions are implicit from the type of relationship defined. So to explicitly define prepositions can sometimes limit the questions that could be asked or could give spurious interpretations. This is generally true with temporal relationships involving times, but it can also be a factor in locative relationships involving places. These matters will be covered in "Prepositions between nouns" on page 51.

5.9 Classifying entities

You classify an entity to determine what types of relationships can be made with. However, before you can define relationships, you must also define a term and part of speech for the concept. The principal exceptions to this rule are names and identifiers, described in "Defining, naming, and identifying table entities" on page 34, and units of measure, described in "Entities referring to numerical data" on page 72.

When classifying concepts, use one of the predefined classes in Ergo or specify one of the types of concept that you have already defined. If you do not use one of the predefined classes, see "Creating noun subclass entities" on page 53 for a description of how to handle these subclasses.

Ergo has 37 predefined classes, listed in Table 1 on page 39. You can use any of these classes to define a column or table except:

- Event. Used for verbs, see "Creating verbs" on page 67.
- Property. Used for general adjectives, see "Creating adjective subclasses" on page 55.
- Qauantified_property, unit_of_measure and its subclasses. See "Quantified_property and unit_of_measure entities" on page 74 for when to use, these types. This does not mean that the *subclasses* of quantified_property, such as age and value, should not be used for tables and columns; this is quite normal for many types of numerical value.

CUSTOMIZATION TOOL USER'S GUIDE

Defined classes
Thing
<ul style="list-style-type: none">• Address• Event• Identifier• Institution• Material_thing• Name• Number<ul style="list-style-type: none">-Quantity• Person• Place• Property<ul style="list-style-type: none">-Quantified_property-Age-Area-Depth-Duration-Height-Length-Price-Size-Value-Volume-Weight-Width• Time<ul style="list-style-type: none">-Date-Period-Timestamp• Unit-of-measure<ul style="list-style-type: none">-Currency-Unit_of_measure_age-Unit_of_measure_area-Unit_of_measure_duration-Unit_of_measure_length-Unit_of_measure_volume-Unit_of_measure_weight

Table 1. Defined classes.

While you can define more than one class for a concept, you almost never need to do this in practice. Most particularly, do not specify multiple classes in these cases:

- The classes are in the same hierarchical chain of subtypes. Once you have defined a term for the concept, whatever relationships you can define for a class are passed on to all the subtypes of that class down through the hierarchy. For example, all subclasses inherit the ability to make possessive relationships from class thing.
- If the classes involved are in the group of five that can take locative relationships, described in "Locative classes" on page 40. For example, university could be classified as both an institution and a place. However, there is no benefit in doing this.

The class you choose for an entity determines what types of relationship can be made with these entities. The remainder of this section describes the relationships that are available for each group of classes.

CUSTOMIZATION TOOL USER'S GUIDE

5.9.1 Thing class

Classify your entity as a subclass of thing if it represents something that is abstract or intangible, such as project or promotion. You can also use this class whenever none of the other predefined classes is suitable.

There are no special relationships for entities classified as thing other than the possessive relationship, described in "Possessive relationships;" on page 44 and the associated relationship, described in "Associated relationships" on page 64.

5.9.2 Locative classes

Five classes can all have a position and so can enter into locative relationships, described in "Locative (place) relationships" on page 45. The differences between them are:

Address	Represents all or part of the address of something.
Institution	Represents an organization, or a group of people considered as a group and not as individuals.
Material_thing	Represents a material object that is not a person.
Person	Represents an individual person, such as <i>employee</i> . If the entity represents a group of people, such as <i>department</i> , classify it as a subclass of institution.
Place	Represents the physical location of something.

5.9.3 Temporal classes

Three classes that can enter into temporal relationships, described in "Temporal (time) relationships" on page 47, are:

Time	Represents values with the SQL data type TIME.
Date	Represents values with the SQL data type DATE.
Timestamp	Represents values with the SQL data type TIMESTAMP.

5.9.4 Numerical classes

These classes can have a measured-by relationship pointing to them:

Number Represents values with numerical data types that can be compared with other numbers, but which cannot be used for arithmetical statements, for example, the job level of an employee.
If *job level* is a subclass of number, users can ask:

Is Smith's job level higher than Wheeler's job level?

Users cannot ask questions with arithmetic expressions, such as:

What is the total job level of employees?

Quantity This is used when the concept denotes a count of something and can thus be used in arithmetical statements; for example, the number of units ordered or the population of a country. In addition to measured-by relationships, you can make counted-by relationships with this class. Because these involve the definition of a new indirect entity, they are

CUSTOMIZATION TOOL USER'S GUIDE

described in "Counted_by relationships" on page 72.

Quantified_property This is used when the numerical value is measured by a unit of measure, such as dollars, feet, or kilograms. Many of these units of measure are predefined and so you just specify which unit the value is measured in. To specify your own units, units of measure, or quantified properties, see "Unit entities" on page 74 and "Quantified-property and unit_of_measure entities" on page 74.

If you select the measured-by relationship, the words described in "Possessive words" on page 45 are automatically available. **Do not** also select the possessive relationship. If you select both the measured-by relationship and the possessive relationship, users may receive duplicate interpretations for their questions.

Subclasses of quantified_property: The twelve predefined subclasses of quantified_property are:

Age	Represents how old someone or something is.
Area	Represents the two-dimensional area of something.
Depth	Represents how deep something is.
Duration	Represents how long something lasts.
Height	Represents how high or tall something is.
Length	Represents how long something is (in space, not in time).
Price	Represents how much something costs.
Size	Represents how big something is.
Value	Represents how much something is worth.
Volume	Represents the three-dimensional volume of something.
Weight	Represents how light or heavy something is.
Width	Represents how wide something is.

The units of measure and the units that can be related to these quantified-properties are given in Table 2 on page 42.

Quantified properties and units of measure		
Quantified property	Unit of measure	Units
price, value	currency	cent, dollar, penny, pound (£), krona
age, duration	unit_of_measure-duration	century, day, hour, minute, month, second, year
depth, height, length, width	unit_of_measure_length	centimeter, foot, inch, kilometer, meter, mile, millimeter, yard

CUSTOMIZATION TOOL USER'S GUIDE

Area	unit_of_measure_area	acre, square_centimeter, square_kilometer, square_meter, square_mile
Volume	unit_of_measure_volume	cubic_centimeter, cubic_meter, gallon, liter, pint, quart
Weight	unit_of_measure_weight	gram, kilo, ounce, pound (lb), ton
Size	all units of measure	all units

Table 2. Quantified properties and units of measure.

5.10 Specifying terms for entities

The terms that you link to the entities are the words that the user will use to access information in the database. If you specify more than one term, the terms are treated as synonyms. You can have as many terms as you want.

If you specify more than one term for an entity, the query interface only uses the first one to generate its interpretation of the question. So ensure that the first term of all entities is different. For example, if you have two entities with the first term *employee*, then the question *who are the employees* will give you two identical natural language interpretations: *find employees*, even though the SQL statements are different. However, if you make *employee* the second or third term of the entities and make the first term something else, like *headquarter employee* and *branch office employee*, then you will get two natural language interpretations:

find headquarter employees
find branch office employees

5.10.1 Types of compound terms

Sometimes you must define a compound term consisting of more than one word. This section describes the differences between them. Some of these are only applicable when you create a new entity, rather than when you define a concept that relates to a table or column in the database. However, all options are included here for completeness. The most commonly used types are:

- **SINGLE NOUN.** Use this form if you want to treat the compound term as a single noun. The term will then be treated as a unit in questions; the user must enter exactly the same number of blanks between the words as are defined in the term. With the other compound terms, users can type any number of blanks between the terms because they are treated as separate but related terms by the natural language engine. Example:

If the term is *Californian location* and the users will always use both terms together, use this option. The term can then only be used in questions such as: *List the Californian locations*.

- **ADJ + NOUN.** Use this form if you want to use the adjective both together with the noun and separated from it. This is used to define a term for an implicit adjective entity, as described in "Creating adjective subclasses" on page 55. Example:

List the plants that are profitable
List the profitable plants.

Note: If you want to use adjectives in questions without the nouns that they modify, you must define an explicit adjective, as described in "Creating adjective subclasses" on page 55.

CUSTOMIZATION TOOL USER'S GUIDE

- ADJ + NOUN +NOUN. Use this form if the compound term consists of an adjective and two nouns. Example: Local sports club or positive trade balance

The other forms of compound term, which are rarely needed, are:

- NOUN + NOUN. Use this form if your compound term consists of two nouns, and you want to use the plural form of one or both of them. Example: Database administrator
- ADJ + NAME. Use this form if your compound term consists of an adjective and a proper name. Example: North America
- NAME + NOUN. Use this form if your compound term consists of a proper name and a noun. Example: Texas customer
- NOUN + NAME. Use this form if the compound term consists of a proper name and a noun. Example: Captain Smith

5.11 Defining relationships

Some questions can be answered by just defining terms for individual columns. However, to use the full potential of Ergo, you must define relationships between entities.

In a database, these relationships are implicitly defined because columns are collected together in a table. However, in natural language questions, these relationships must be explicitly defined, to a greater or lesser extent.

Relationships between words in an English sentence are determined in many ways, most particularly with verbs and prepositions, such as of, in, by, and so on. The principal exception to this is the possessive relationship, which can often be denoted either by the preposition of, as in salary of employee, or with the genitive case, as in employee's salary. This possessive relationship can also be denoted with the adjacency of words, such as department manager, or with the verb *have*, as in *the department has a manager*.

You do not need to define many of these prepositional relationships explicitly; they follow naturally from the types of relationship that you define. These are described in this subsection. However, some prepositional relationships must be defined explicitly, especially when verbs are involved in the relationship. These are described in "Prepositions between nouns" on page 51.

The types of relationships you can define for nouns are:

- Identifier relationship
- Name relationship
- Refer-to relationship
- Associated relationship
- Possessive relationship
- Four locative (place) relationships
- Three temporal (time) relationships
- Duration relationship
- Measured-by relationship
- Prepositional relationships
- Counted-by relationship

The first three relationships are described in "Defining, naming, and identifying table entities" on page 34, the fourth in "Associated relationships" on page 64, and the last is described in "Counted-by relationships" on page 72 because it requires a new entity to be defined. The remaining noun relationships are described here.

CUSTOMIZATION TOOL USER'S GUIDE

The type of relationship that you can define between entities is determined by the types of concept involved in the relationship and whether terms have been defined for the entities. These potential relationships are inherited by the subtypes of each class of entity. That is, the relationships that can be defined for any particular level of concept can also be defined for all subtypes of the concept.

While all these relationships can be defined between any table and column, define the relationships within a single table before defining inter-tabular relationships. Most of the intra-tabular relationships will be between the table and the columns in the table. However, if the table contains a column that is related to a candidate key that is not the primary key, you may need to define relationships between these columns.

Note that relationships have a direction. One direction is determined by the entity-attribute relationship between tables and columns. However, the entity-attribute relationship can be viewed in reverse; that is, between the column and the table. For example, we can write both manager of department and department of manager in a possessive relationship. However, while employees have salaries, salaries do not have employees. Be careful, when selecting the relationship you want, to note the direction of the relationship.

When defining relationships between entities, many prepositions are implicitly defined and can be used when asking questions. These prepositions are listed under each of the relationships in this section. Do not explicitly define any of these prepositions when defining relationships. If you need to explicitly define prepositions in relationships, see "Prepositions between nouns" on page 51.

For example, you may want to ask questions about orders:

on a date
of a product
as a salesrep

The prepositions in the first two of these are automatically available with suitable definitions of relationships. You need define only the last of these prepositions explicitly.

5.11.1 Possessive relationships

Whenever you have defined terms for two entities, you can define possessive relationships in both directions between the entities. For example, if you defined a possessive relationship from the table entity, employee, to the column entity, salary, you can ask questions such as:

What are the salaries of the employees?

Here are some relationships you can define:

What does the department have? employee
What does the employee have? department

These are possessive relationships that let you ask:

What is the department of John?
Who are the employees of the marketing department?

Possessive words: When you link a noun entity (for example~, project) to another entity, these relationships may be available:

- If the other entity is a noun (with any class):

What does the project have?

CUSTOMIZATION TOOL USER'S GUIDE

- If the other entity is a name:

What is the name of the project?

If you select any of these relationships, these prepositions are available:

for
of
per
with
without

Possessive relationships also let you use possessive refer-back pronouns. For example:

List the employees and their salaries.

These refer-back pronouns are:

his, him (if *he* is valid for the noun)
her (if *she* is valid for the noun)
its (if *it* is valid for the noun)
their

Note: Non-possessive refer-back pronouns can be used regardless of what relationships you have defined. For example, you do not need to define any relationships for this pair of questions to be answered:

*List managers
Where do they work?*

Example: If *What does the employee have?* is the relationship between *manager* and *employee*, users can ask questions such as:

*Who is the manager of Jones?
Who is Smith's manager?
List clerks and their managers.*

5.11.2 Locative (place) relationships

There are four locative relationships that you can link to entities in the five locative classes listed on page 40 and any of their subclasses:

- Where (locative place)
- From where (locative source)
- To where (locative goal)
- Via which intermediate stages (locative path)

Some examples of these relationships are given here. For locative place expressions, you select:

Where is the employee? department

This lets you get answers to questions such as:

*In what department is John?
and
Who is in the marketing department?*

CUSTOMIZATION TOOL USER'S GUIDE

The relationship:

From where does the Employee come? Department

will let you answer questions such as:

From what department is John?

and

Who is in the marketing department?

If you select one of these relationships, several prepositions and other words are automatically available. For a list of these words, and an example of how they are used, see "Location words" on page 46.

Example: In EPEDEMOE, *location* has this relationship:

Where is the department? location

5.11.3 Location words

The prepositions that are implicit for locative relationships depend on the type of relationship; each relationship has its own set of prepositions.

Where: If you select a where relationship, these prepositions are available:

above	before	near
across	behind	next to
against	below	off
ahead of	beneath	on
among	beside	opposite
apart from	beyond	outside
around	down	over
at	in	under
away from	inside	within

These words are also available:

all over	everyplace	someplace
anyplace	everywhere	somewhere
anywhere	nowhere	where
elsewhere		

Example: If you select the relationship *Where does someone work?*, users can ask questions such as:

Who works in head office?

Who works under Pacific department?

List staff who work anywhere.

From where: If you select a from where relationship, these prepositions are available:

apart from

away from

CUSTOMIZATION TOOL USER'S GUIDE

from
starting
starting from

The word *where* (if the question also contains one of the above prepositions) is also available.

Example: If you select the relationship *From where does something move?*, users can ask questions such as:

Where was part 3001 moved from?
List parts that were moved starting from the warehouse.

To where: If you select a to where relationship, these prepositions are available:

above	down	over
across	in	till
against	into	to
ahead of	near	toward
among	next to	towards
around	off	under
before	on	until
behind	onto	up to
below	out	

The word *where* is also available.

Example: If you select the relationship *To where does something move?*, users can ask questions such as:

Where was part 3001 moved?
List parts that were moved towards the warehouse.

Via which intermediate stages: If you select a via relationship, these prepositions are available:

across
along
by
by way of
through
via

Example: If you select the relationship *Via which intermediate stages does someone travel?*, users can ask questions such as:

Who travels through France?
By way of which countries does Smith travel?

5.11.4 Temporal (time) relationships

There are three temporal relationships that you can link to entities with a class of time or a subclass of time. They are:

CUSTOMIZATION TOOL USER'S GUIDE

- When (temporal place)
- Since when (temporal source)
- Till when (temporal goal)

If you link a time entity to a noun (for example, project), the time entity can represent the answer to any of these questions:

When is the project?

Since when is the project?

Till when is the project?

5.11.5 Time words

The prepositions that are implicit for temporal relationships depend on the type of relationship; each relationship has its own set of prepositions.

When: If you select a when relationship, these prepositions are available:

after	during	in
at	for	of
before	from	on

These words are also available:

always	sometime	tomorrow
anytime	the day after tomorrow	when
never	the day before yesterday	yesterday
now	today	

Example: If you select the relationship *When does someone travel?*, users can ask questions such as:

Who travels on 12/03/04?

Did Smith travel yesterday?

Since when: If you select a since when relationship, these prepositions are available:

after
from
since
starting

The word *when* (if the question also contains one of the above prepositions) is also available.

Example: If you select the relationship *Since when is the project?*, users can ask questions such as:

List projects starting in June.

Till when: If you select a till when relationship, these prepositions are available:

CUSTOMIZATION TOOL USER'S GUIDE

before	till	until
for	to	up to

The word *when* (if the question also contains one of the above prepositions) is also available.

Example: If you select the relationship *Till when is the project?*, users can ask questions such as: >

List projects up to June.

5.11.6 Duration relationship

The duration relationship is a special kind of measured-by relationship that you can use with entities classified as duration. This lets users ask questions such as:

How long has John worked?

Duration words: The prepositions that are available with the duration relationship are:

during
for
in

These words are also available:

new	old	short
long	recent	young

Example: If you select the relationship *How long does someone work?*, users can ask questions such as:

For how long has Smith worked?

List employees who have worked for 10 years.

5.11.7 Measured-by relationship

You can define a measured-by relationship with any entity classified as a number, its subtype, quantity, quantified-property, and its subtypes. You define the measured-by relationship by selecting the choice:

What is order measured by? quantity

In the define relationships dialog box.

When users ask questions about a number entity, they can use these words:

big
great
high
large
small

This relationship lets users ask questions such as:

CUSTOMIZATION TOOL USER'S GUIDE

List orders greater than 300.

If users want to ask questions like the following, you must define a measured-by relationship to a quantified_property that has a unit defined:

How big are the orders?

List the top three orders.

List the three biggest orders.

List the biggest order.

If you have an entity with the quantified_property size, then the measured-by relationship also lets these questions be answered:

What is size of the orders? (with no unit of measure)

What is size of the biggest order? (with unit of measure)

The question words listed in Table 3 on page 50 are available when you define measured-by relationships to entities classified as quantified property

Question words available with measured-by relationships		
What is the...	How...	How many...
Age	old, young	centuries, days, hours, minutes, months, seconds, years
Area	-	acres, square centimeters, square kilometers, square meters, square miles
Depth	deep, shallow	centimeters, feet, inches, kilometers, meters, miles, millimeters, yards
Height	high, tall, long, short	centimeters, feet, inches, kilometers, meters, miles, millimeters, yards
Length	long, short	centimeters, feet, inches, kilometers, meters, miles, millimeters, yards
Price	cheap, expensive, high, low	cents, dollars, pennies, pounds
Size	-	-
Volume	-	cubic centimeters, cubic meters, gallon liters, pints, quarts
Value	cheap, expensive, high, low	cents, dollars, pennies, pounds
Weight	heavy, light	grams, kilos, ounces, pounds, tons
Width	broad, narrow, wide	centimeters, feet, inches, kilometers, meters, miles, millimeters, yards

Table 3. Additional question words available with measured-by relationships.

CUSTOMIZATION TOOL USER'S GUIDE

Note: British English spellings centimetre, kilometre, metre, and millimetre are also supported.

5.12 Prepositions between nouns

When defining prepositional relationships between nouns, you should also consider what verbs you are using and whether the prepositions should be attached to the verbs rather than the nouns, Defining noun prepositional relationships sometimes simplifies the definition of verbs, but the language coverage of the preposition is affected. See Table 4 on page 123 for a full list of the prepositions you can use.

As examples of noun prepositional relationships, if sales representatives take orders for products from customers, you can define the prepositions for the noun order with this syntax:

1. Define the noun complement order *from something* and *order for something*.
2. Define the syntax for the noun order *from something for something*.
3. Define the relationship between order and customer by selecting:
What is someone/something the order from? customer
4. Define the relationship between order and product by selecting:
What is someone/something the order for? product

When you come to define the verb take, you merely need to define the subject and the object for the verb. That is, the sales representative is the subject of the verb and the order is the object. The other relationships will automatically be derived from the prepositional relationships involving the noun order.

With this customization, the user could now ask the question:

Who took orders from Smith? (where Smith is a customer.)
Who took orders for hammers? (where hammers are products.)

Note that there are some limitations when prepositions are linked to nouns rather than verbs. When a preposition is linked to a noun, the prepositions in the question must be syntactically related to the noun and not the verb. If the preposition is only linked to the noun then users cannot ask:

For what (products) were the orders taken?
What were the orders taken for?
From whom were the orders taken?
Who were the orders taken from?

For these questions to work, the prepositions must be linked to the verb. However, if they are linked only to the noun, users can ask:

For what products were the orders?
What were the orders for?
From whom were the orders?
Who were the orders from?

An alternative to this customization is to define customer as a person or institution and then choose the *from where* relationship. This permits:

Where were the orders taken from? (relationship with verb) *Where were the orders from? (relationship with noun)* and all the examples given above. If the relationships are defined for both the verb and the noun (direct object), some multiple interpretations may occur, but the advantages in terms of coverage could outweigh the disadvantages.

CUSTOMIZATION TOOL USER'S GUIDE

When defining temporal relationships do not specify explicit prepositions; an explicit preposition should never be defined between a noun and a noun classed as time. *

For example, do not specify the preposition *on* linking interview and interview date like this:

What is the interview on? Interview date

If you do, users cannot get answers to questions involving these entities. Instead, use the *when* temporal relationship, which will give the preposition implicitly. For example, specify:

When is the interview? Interview date

Users could then ask questions such as:

When are the interviews?

Are there any interviews in June?

List interviews on July 30th, 2001.

CUSTOMIZATION TOOL USER'S GUIDE

6 DEFINING SUBSETS OF THE DATA

The previous chapter described how you define concepts that have a direct relationship to particular tables and columns in the database. This chapter explains how you define concepts that are subclasses of these main concepts. Sometimes these subclass concepts have a direct relationship to one or more columns in the database and so an entity will already exist for these concepts. In other situations, you will need to create a new entity, together with its associated SQL statement.

The three ways of creating subclass concepts that denote a subset of the data are:

Noun subclass

Define an existing or a new noun entity as a subclass of an entity that you have defined.

Instance

Define a new entity as an instance of an entity that you have defined.

Adjective subclass

Define an adjectival compound term or explicit adjective entity.

6.1 Creating noun subclass entities

You define an entity as a noun subclass of one of your defined entities if it represents a subset of the data values of the super-class. For example, programmers are a subset of all employees and businesses are a subset of all customers.

The terms for these subclasses will sometimes be single words, but more often they will have multiple words, with the first qualifying the main concept. Examples are department managers and retired employees. In these situations you define the term as a single noun unless you want to treat the qualifier as an adjective. This is described in "Creating adjective subclasses" on page 55.

The two types of noun subclass are:

Complete column

A subclass may refer to a column of data values, all of which are subsets of another column of data values. These columns are normally defined as inclusion dependencies in database mode. The SQL for these subclasses is generated by the Customization Tool.

Examples of this type of subclass in EPEDEMOE are department managers and ordered products.

Subset of column

A subclass may refer to a subset of data values in a single column based on some quality, property, or criterion. Here you must define a new entity with an SQL statement that specifies the selection criterion for the entity. If the super-class is a table entity, the selected column in the SQL statement must be the identifier of the table, unless the table has a multicolumn key. In these instances, you must take another approach, which is described in "Composite identifiers" on page 62:

Sales persons and tools are examples of this type of subclass in EPEDEMOE. They are subclasses of employee and product, respectively. The SQL statements for these two entities are:

```
SELECT X1.ID FROM EPE.STAFF X1 WHERE X1.JOB='SALES'
```

and

CUSTOMIZATION TOOL USER'S GUIDE

```
SELECT X1.PRODNUM FROM EPE.PRODUCTS X1 WHERE X1. PRODGRP='TOOL'
```

When defining two or more subclasses of user-defined classes, note that the Customization Tool assumes that these subclasses are Intersecting unless you specify otherwise. For example, managers and clerks as subclasses of employee are defined as intersecting. If employees cannot be both managers and clerks, you must select disjoint when defining these subclasses.

The important point to note about subclasses is that relationships that exist between a super-class and other entities are generally inherited by the subclass. For example, in EPEDEMOE, products have prices, so tools also have prices. There is no need to define a relationship between tools and product price. You can then ask questions like:

What is the average price of tools?

The natural language interpretation for questions that involve an inherited relationship shows the relationship to the super-class with the subclass in parentheses. For example, the natural language interpretation for the above question is:

Find the average prices of products (that are tools).

However, if the relationship from the super-class involves an explicit preposition, the relationship is not automatically inherited. For the subclass to inherit the relationship, you need to specify the same preposition for the subclass.

When the new entity that you define is a subclass of a table entity, as in the examples of sales persons and tools above, you normally define just one entity and include the SQL statement with that entity. The names relationship is inherited just like any other.

However, you can also create a complete structure that matches the relationships of the table entity to its identifier. In this case you would define the subclass entity without an SQL statement, define another entity with the SQL statement, and connect them with an identifier relationship. This entity does not need a term, but you do need to connect it to the subclass entity in the normal way.

This more complex way of customizing subclasses of table entities has no advantages with single column identifiers. But you use this method with composite entities, described in "Composite identifiers" on page 62.

6.2 Creating instance entities

You create instance entities in a similar manner to the second type of noun subclass entity. The difference between them is that subclasses refer to a general type of something, like a product type, while instances refer to a particular instance of a type. Products that have individual serial numbers are examples of instances, although you would not normally consider defining each of these as instances in a model.

The most common usage of instance entities are natural language representations of coded data values, such as department codes, like ADM and FIN for administration and finance, or M and F for male and female. If the full meaning of these codes is not stored in a database table, you can define an instance entity to represent the code.

As with any other entity, you can define more than one term for an instance. So you can define synonyms for the codes in the database, such as HQ and Headquarters.

When the rows of the table represent the instances being modeled, the selection criterion for the instance entity refers to just one row in a table. This might well be the case with a department table containing department codes but not names. In these cases, the column (or columns) whose values you are modeling is a primary or candidate key.

When the number of instances is comparatively small, it is often helpful to the questioner to define

CUSTOMIZATION TOOL USER'S GUIDE

instances when the terms in the instances are the data values themselves, rather than codes. This has been done in EPEDEMOE, where the department names and locations have been defined as instances even though the full name is stored in the database.

The benefit of defining instances in these cases is that the performance of the natural language engine is improved. Because such concepts as department names and locations are defined rather than undefined terms, the engine can analyze the question more directly.

Defining instances in this way also reduces the possibility of multiple interpretations of questions. For example, in EPEDEMOE, managers manage both departments and projects. If you asked this question with no defined instances:

Who manages Pacific?

you would get two interpretations because the engine does not know whether Pacific is the name of a department or an identifier for a project. If, however, you define Pacific as an instance of department name, then the engine knows that Pacific is not a project and will not give that possible interpretation.

Note: Because instances refer to particular things or notions, they are automatically defined as disjoint; you cannot change this.

6.3 Creating adjective subclasses

An adjective is a word that modifies a noun. In Ergo, an adjective describes a quality or characteristic of another entity. It has the effect of reducing the set of data values that relate to the entity. For example, in EPEDEMOE, electrical and senior are adjectives that modify products and employees, respectively. Electrical products and senior employees are subsets of all products and employees, respectively.

You define adjective subclasses in a similar manner to noun subclass entities, but there are some differences depending on how you define the adjective. You can define adjectives in three ways in the Customization Tool:

Implicit

An adjective-noun noun compound that is a subclass of the noun entity that it is modifying.

Specific

An explicit adjective entity that is a subclass of the specific noun entity that it is modifying.

General

An explicit adjective entity that is a subclass of the defined class property. You can then link this entity to multiple entities through implicit adjective entities that are subclasses of both the general adjective entity and the noun entities that they are modifying.

6.4 Implicit adjectives

You define an implicit adjective when the adjective is associated with just one noun, and users will always ask questions containing both the adjective and the noun that it is modifying. Because the adjective only exists implicitly, a noun icon rather than an adjective icon is displayed by the Customization Tool. Also, you can define any relationship that is available for noun entities, described in "Defining relationships" on page 43.

For example, in EPEDEMOE, the adjective electrical is defined as an implicit adjective, electrical_product, as a subclass of product. This customization lets you ask these questions:

List electrical products.

What products are electrical?

However, you cannot ask *What is electrical?* To ask this question, you define a specific adjective.

CUSTOMIZATION TOOL USER'S GUIDE

To define an implicit adjective:

1. Create a noun entity as a subclass of the noun entity that the designated adjective modifies, for example, product.
2. Define what other subclasses this subclass is disjoint with. For example, electrical is disjoint with mechanical but intersecting with expensive. You can ask List expensive electrical products, but you should prevent the user asking list electrical mechanical products by defining mechanical_product and electrical_product as disjoint.
3. Give the subclass entity an adjective-noun compound term, for example, electrical goods. If you also define goods as a term in the super-class entity, you can use the adjective with any of the terms in the super-class, such as product, and any intersecting subclasses of the super-class entity.
4. Define an SQL statement for the entity. For example:
`SELECT X1.PRODNUM FROM EPE.PRODUCTS X1
WHERE X1.PRODGRP = 'ELECTRICAL'`

6.5 Specific adjectives

You define a specific adjective when the adjective is associated with just one noun, and users want to ask questions with the adjective only as well as questions containing both the adjective and the noun that it is modifying. You should try this form of the adjective first, unless you want to use relationships that are available for nouns but not adjectives.

For example, in EPEDEMOE, the adjective senior is defined as a specific entity as a subclass of employee. This customization lets you ask these questions, as with the implicit adjective:

*Find senior employees.
Are there any senior employees?
What employees are senior?
Is Smith a senior employee?*

However, with specific adjectives, you can also ask:

*Is Smith senior?
Who is senior?*

Because the adjective senior is explicitly defined as a subclass of the noun that it modifies, the engine can link it to the noun, even though the noun is not mentioned in the question. You may not get a natural language interpretation in the second example, but you can still send the generated SQL to the database.

To define a specific adjective:

1. Create an adjective entity as a subclass of the noun entity that the designated adjective modifies, for example, employee. The entity will appear in the diagram as a triangle to indicate that it is an explicit adjective.
2. Define what other subclasses this subclass is disjoint with. In EPEDEMOE, the adjective senior is intersecting with all other subclasses of employee, such as clerk, so you do not need to do anything here.
3. Give the subclass entity a term, for example, senior. You can use the adjective with any of the terms in the super-class, such as staff, and with any intersecting subclasses, such as clerks.

CUSTOMIZATION TOOL USER'S GUIDE

4. Define an SQL statement for the entity .For example:
SELECT X1.1D FROM EPE.STAFF X1
WHERE X1.YEARS > 5

This means that senior employees are defined as those with more than five years experience.

6.6 General adjectives

You define a general adjective when the adjective is associated with more than one noun. Even though general adjectives have an adjective icon, they act like implicit rather than specific adjectives; you must include the noun that the adjective modifies in questions.

One benefit of using a general adjective rather than multiple implicit adjectives is that the adjective is stored in the model just once, rather than for each implicit adjective. Another use of general adjectives is when you want to define prepositional relationships that relate to the adjective rather than the noun, as described in "Prepositional relationships" on page 58.

To define a general adjective:

1. Create an adjective entity as a subclass of *property*.
2. Give the adjective entity a term, for example, valuable.
3. Do not define an SQL statement for the adjective.
4. For each noun to which this adjective is to be linked:
 - a. Create an implicit adjective, for example, valuable product, valuable order, and valuable customer.
 - b. Classify each implicit adjective as a subclass of the noun entities, such as product, order, and customer.
 - c. Define an SQL statement for each implicit adjective that delimits the meaning of the adjective.
 - d. Classify the implicit adjective as a subclass of the general adjective.

6.7 Relationships with adjective entities

Explicit adjective entities can have three types of relationship associated with them:

- Locative relationships
- Temporal relationships
- Prepositional relationships

Note that locative and temporal relationships for the super-class of the adjective are normally inherited by the adjective. So you do not need to explicitly define relationships in these cases. However, if the relationship applies to the subclass entity but not to its super-class, then you should define the relationship with the adjective.

6.7.1 Locative relationships

If you link an entity with a locative class to an adjective (for example, available), you could select these relationships:

Where is product available? location
From where is product available? supplier

With other adjectives, such as en route, you could define some of the other locative relationships:

CUSTOMIZATION TOOL USER'S GUIDE

From where is delivery en route? dispatcher
To where is delivery en route? customer
Via which intermediate stages is delivery en route? intermediate location

Other examples of adjectives that you can use in locative relationships are *underway* and *located*, defined as an adjective and not as a verb without a subject.

If you select one of these relationships, the same location words are available as for nouns. For a list of these words, and an example of how they are used, see "Location words" on page 46.

6.7.2 Temporal relationships

If you link an entity with a temporal class to an adjective (for example, *senior*), you could select these relationships:

When is employee senior? seniority date
Since when is employee senior? seniority date

With other adjectives, such as *available*, you could define other temporal relationships:

Since when is offer available? start date
Till when is offer available? close date

Other examples of adjectives that you can use in temporal relationships are *applicable* and *effective*.

If you select one of these relationships, the same time words are available as for nouns. For a list of these words, and an example of how they are used, see "Time words" on page 48.

6.7.3 Prepositional relationships

You can define prepositional complements to accompany the adjective. The available prepositions are the same as for nouns and verbs and are given in Table 4 on page 123.

For example, in EPEDEMOE, the adjective *responsible* has been defined with the preposition *for*. Relationships have then been defined that show:

Department managers responsible for departments
Department managers responsible for projects

To define a prepositional relationship for an adjective:

1. Create a specific adjective as a subclass of the noun that it modifies, for example, department manager.
2. Give the adjective the category adjective and a term, for example, responsible.
3. Define the syntax for the adjective, for example, *responsible for something*.
4. Specify the SQL statement for the entity, for example:

```
SELECT X1.MANAGER  
FROM EPE.ORG X1
```

5. Create the prepositional relationship to the relevant noun entities, for example:

What is someone/something responsible for? department
What is someone/something responsible for? project

CUSTOMIZATION TOOL USER'S GUIDE

Other examples of adjectives that can have prepositional relationships are *experienced in* and *answerable to*.

CUSTOMIZATION TOOL USER'S GUIDE

7 DEFINING MULTICOLUMN ENTITIES

An entity in a conceptual model can refer to only one column in a table in a database, either directly or indirectly. Sometimes, however, you need to refer to multiple columns in a database. You can do this in three ways in Ergo:

Calculated entities

Entities where the column is determined from an expression that is based on one or more columns.

Composite entities

Entities that combine two or more column entities into a single entity.

Associated relationships

Defines entities associated with other entities and displayed with these entities in the answers to questions.

Note: Table entities that have both a name and an identifier act like composite entities, but you do not need to define them as such.

7.1 Calculated entities

Databases rarely store all the information that can be derived from the data in the database; much information is calculated from the stored columns. While users can enter expressions in questions to specify this calculated information, it helps to define calculated entities explicitly in your model.

Examples of calculated entities in EPEDEMOE are income, defined as salary + commission, and order_value, defined as order_qty * product_price.

To define a calculated entity:

1. Create a noun entity with a numerical class, such as value for income and price for order_value.
2. Give the entity terms as normal.
3. Enter an SQL expression for the calculated entity. For example, income is defined as:

```
SELECT (X1.SALARY + X1.COMM) FROM EPE.STAFF X1  
WHERE X1.COMM IS NOT NULL
```

and order-value is defined as:

```
SELECT (X1.QUANTITY * X2.PRODPRICE) FROM EPE.SALES X1, EPE.PRODUCTS X2  
WHERE X1.PRODNO = X2.PRODNUM
```

Note that in this second example you must define a join path because the values in the column expression come from different tables.

4. Define relationships as appropriate to the new entity.

Note: A full description of defining SQL statements in Ergo is given in "Writing SQL statements" on page 126.

CUSTOMIZATION TOOL USER'S GUIDE

7.1.1 Handling null values

When you have columns that can contain null values you need to be careful how you combine them in calculated entities, for the results might not be what the user expects.

For example, the entity commission in EPEDEMOE is defined as the sum of salary and commission. However, commission is only defined for sales representatives; for all other employees, the commission is null. But an expression in SQL containing null values results in a null value; nulls are not treated as zero.

This means that the term income has meaning only for sales representatives. So the questions:

List incomes of managers

List incomes of employees

will give no output for the first question and only list the sales representatives in the second.

You can handle this situation in a number of ways:

1. Define income as a synonym of salary. Users will then receive multiple interpretations. They can then choose income or salary as appropriate to their question.
2. Relate income and commission to the salesreps entity. This will give users interpretations that indicate that the concepts of income and commission apply only to sales representatives.
3. Combine methods 1 and 2.
4. Remove the not null clause from the SQL statement for incomes, and create an associated relationship between income and salary, as described in "Associated relationships" on page 64.
5. More drastically, change the null values in the database to zeroes. But then users could not ask questions about employees who had no commission, such as *List employees without commission*.

7.2 Composite entities

When you want to handle two or more column entities as a single entity, you define a composite entity. A composite entity consists of a sequence of entities, not necessarily from the same table. If the constituent entities come from different tables, you must define a join path between the tables: You can have composite entities for any class of concept.

A composite entity does not have an SQL statement associated with it, but all the entities constituting the entity must have SQL statements. These statements can refer to a single column or be calculated entities, as described in "Calculated entities" on page 60. However, the SQL statements in composite entities cannot have SQL column functions in them, such as AVG. Also, a component entity of a composite entity cannot be another composite entity.

You can define four main types of composite entity:

Composite identifiers

Denote multicolumn keys.

Composite names

Denote multicolumn names.

Composite columns

Denote other multiple columns that you want to handle as a single column, such as address columns.

CUSTOMIZATION TOOL USER'S GUIDE

Composite reports

Denote multiple columns that you want to output together as a report.

You define a composite entity during the classification step of defining the entity. You select the option consists of. You are then given a list of entities that you can select as the components of the composite entity. The order in which you select these entities is significant. You can have up to 255 components in a composite entity.

You can classify and give terms to composite entities and define relationships to them. You can also use the terms representing the composite entity, but if you want to use a data value from the composite entity as an undefined term in a question, you should give a word for each component of the composite entity if you want to avoid multiple interpretations.

For example, in EPEDEMOE, the names of the interviewees are defined as a composite name consisting of the first and last name columns. So if you ask the questions:

When was reid interviewed?
Who interviewed john?

you get multiple interpretations because the engine does not know whether reid and john are referring to the first or second names.

To avoid the extra readings, you can denote the missing columns with % (global search character), or you can specify the composite terms in full, like this:

When was % reid interviewed?
Who interviewed john stanley?

7.3 Composite identifiers

You need a composite identifier for each table having a multicolumn primary key.

To create a composite identifier:

1. Create a new entity and give it a name.
2. Classify it as an identifier.
3. Select its component entities.
4. Link it to the table entity that it identifies.

Generally, composite identifiers, like single-column identifiers, need no terms; the terms used by the table entity relate to the identifier.

The Customization Tool generates SQL statements autocratically for the component entities of the composite entity. You do not normally need to change these. You also do not normally need to classify and give terms to the component entities, unless you want to reference them individually in questions. Here, you can treat these entities as single entities in the normal manner.

If you want to define subclasses of table entities with multicolumn identifiers, you must define a composite entity with exactly the same structure as the super-class entity. How you do this depends on whether the set of data values in the subclass comes from a complete column or not.

Complete column

CUSTOMIZATION TOOL USER'S GUIDE

When the subclass refers to complete columns in a table, you create a new composite entity just like for the super-class. You also define it as a subclass of the super-class composite entity.

Subset of column

When the subclass refers to a subset of columns in a table, you create a new composite entity as for complete columns. You must also define the search-condition that defines the subset of the columns in the SQL statement of one of the component entities of the composite entity.

7.4 Composite names

You need a composite name for each table with a multi-column name.

To create a composite name:

1. Create a new entity and give it a name.
2. Classify it as a name.
3. Select its component entities.
4. Link it to the table entity that it names.

Generally, composite names, like single-column names, need no terms; the term *name* is automatically available as a reference to the name. With composite names, you can also use the terms *first name* and *last name* to specifically refer to the first and last columns in a multicolumn name. There is also no need to define these terms in the model. If you use these terms with a single-column name, the engine will ignore the references to first and last.

For example, if an employee table has a multicolumn name, unlike the STAFF table in EPEDEMOE, then you could ask these questions:

List last names of employees.
List first names of salesreps.

A name of a subclass whose super-class has a composite name will also be a composite name. But you do not have to create the composite name if the super-class entity has an identifier, composite or not, you get it automatically. In these cases, the natural language engine will find the name from the identifier.

If the table does not have separate columns for identifier and name, then you can define the multiple columns as both an identifier and a name to make the terms *name*, *first name*, *last name*, *id*, and *identifier* available, as with single-column entities, described in "Defining terms for names and identifiers" on page 36.

7.5 Composite columns

Several column entities may together represent an attribute of a main concept. For example, an employee table may have the address columns street, city, and state. Entities representing these columns may together constitute an employee_address of employees.

If you want to ask:

List the employees and their addresses.

you can define a composite column in this way:

CUSTOMIZATION TOOL USER'S GUIDE

1. Create a composite entity for employee-address.
2. Specify the term, address.
3. Define a possessive relationship between employee and employee-address.

You do not need to define classes and terms for the component entities unless you want to refer to them individually. If you do, you can define the component entities as single entities in the usual manner. After you have defined possessive relationships to these entities, you could then ask:

*List the cities of salesreps.
Sort managers by state and city.*

Similarly, employees might have a training history consisting of internal courses and external courses they have attended. You could create a composite entity, training_history, consisting of internal_course and external_course. Then you could ask:

List smith's training history.

7.6 Composite reports

Because the SELECT clause of the SQL statement for a particular entity can result in only one column, if you want to create a report consisting of multiple columns, you must create a composite entity for the report. The component columns of the report could be individual columns or calculated columns.

For example, in EPEDEMOE, a composite entity, interview, has been defined, consisting of each interviewee's last name, first name, identifier, and the hiring decision. So the question:

List interviews.

gives a complete report on the interviews with the columns specified.

In another example, you may want an income report composite entity, consisting of the calculated entities, income and income_after_tax, both of which give information about the gross and net income of salesreps. Income could have an SQL statement:

```
SELECT X1.SALARY + X1.COMMISSION
```

and income after tax could have:

```
SELECT X1.SALARY + X1.COMMISSION -X1.TAX
```

To let the user produce a report consisting of these columns, you must define a composite report with its constituent entities to represent the report.

Because you can select entities from different tables in a composite report, report entities have a similar effect to database views. However, the SQL syntax is more limited in the Customization Tool than in SQL view statements. Most particularly, component entities cannot contain SQL statements with column functions COUNT, SUM, AVG, MIN, and MAX.

7.7 Associated relationships

In general, Ergo will include only those columns in a report that are explicitly referenced in questions. One way to generate reports with additional columns is to use composite entities, as described in "Composite reports" on page 64. However, when asking questions involving composite entities, users

CUSTOMIZATION TOOL USER'S GUIDE

sometimes need to be aware of the constituents of the composite entity as explained on page 61.

To avoid this situation, you can define associated entities that are to be displayed with single or composite entities through associated relationships. These associated entities should be functionally dependent on the entity with which they are associated. That is, you should associate column entities to table entities, but not the other way round.

For example, if you always want to display employees' salaries whenever employees are mentioned in a question, you can define an associated relationship between employee and salary. Unlike composite entities, you do not need to create a new entity when defining an associated entity; you just define an associated relationship:

1. Define a relationship between the two entities involved by dragging or clicking on the connector between the entity icons.

2. Select:

What is output with employee? salary
from the list box of relationships that appears.

The associated relationship is inherited by the subclasses of the associated entity. For example, if you define an associated relationship between employee and experience, the experience column will be output with any of these questions:

List employees.

List clerks.

List experience of salesreps.

Because experience is functionally dependent on employees, the engine will display the sales representative's name and identifier once only, instead of twice as implied by the model.

If a super-class and a subclass both have associated entities, you will normally get both associated entities in reply to questions involving the subclass. For example, if you associate experience with employee and department with manager, and ask:

List managers.

you will get a report containing both department and experience of managers. If an entity is a subclass of two different classes which both have associated relationships, both these relationships will be inherited by the subclass.

However, if you have defined different associations to subclasses of a super-class, you will not always get these associations when asking questions that combine these subclasses. For example, if you had another subclass of employee, programmer, and associated programming language with programmer, you would only get the experience of employees when asking this question:

List managers and programmers.

This is because questions like this use the SQL UNION clause to combine the list of managers and the list of programmers. If their different associated entities were also included in the output, the two SQL statements relating to managers and programmers would then be incompatible. The associated entities are therefore not included in the report.

However, if managers and programmers were intersecting concepts, then you would get both experience and programming language when asking this question, which does not generate an SQL UNION statement:

Who is both a manager and a programmer?

Furthermore, associated relationships are not transitive. For example, if you associate department

CUSTOMIZATION TOOL USER'S GUIDE

with employee and location with department, you will not get locations in your report when you ask:

List employees.

You can associate an entity to a composite entity, but if you associate an entity to a component of the composite entity, you will not receive the association for questions that refer to the composite entity as a whole. The association is only output when the question refers to the constituent entity itself.

CUSTOMIZATION TOOL USER'S GUIDE

8 DEFINING OTHER INDIRECT ENTITIES

Most of the entities that you have defined so far have a direct relationship to one or more columns in the database and so have an SQL statement associated with them. The principal exceptions to this are table and composite entities.

This chapter describes three other indirect entities that you can create in your conceptual model:

- Verbs and their relationships
- Counted-by entities and relationships
- User-defined quantified properties, units of measure, and units

8.1 Creating verbs

A verb is a word or a group of words that usually express action, occurrence, or a state of being. In Ergo, verbal concepts let you ask questions like:

Where does John work?
What did the customers buy?

To define a verb, you create a new entity. But this entity is quite different from most other entities in the conceptual model. In particular, verbs do not have SQL statements associated with them.

The easiest way to think of a verb is to think of it as a relationship rather than an entity. It is an explicit relationship, requiring its own hexagonal icon in the diagrammatic representation of the conceptual model.

To create a verb:

1. Create an entity.
2. Give it an entity name.
3. Classify it as event.
4. Give it a term, selecting the verb option.
5. Select the syntax option and specify its verb complement.
6. Continue with the syntax option and specify its prepositional complement.

After creating the verb entity, define its relationship to other entities in the model.

8.2 Defining the verb complement

The verb complement window lets you select the proper syntax to be used corresponding to the verb. The verb complement describes what objects can follow the verb. Verbs that have direct objects are called transitive verbs, and those verbs that do not require a direct object are called intransitive verbs. For example:

Verb with no object (intransitive)

Who lives?
People live in towns.

Verb with a direct object (transitive)

Who assigns what?
Managers assign projects

Verb with direct and indirect object (ditransitive)

CUSTOMIZATION TOOL USER'S GUIDE

*Who gives what to whom?
People give books to people.
People give people books.*

Verb with two direct objects (double accusative)

*Who named whom whom/what what?
Who named the modules DRGS?*

Which verb syntax you choose determines what relationships you can make to the verb, and how many prepositions you can attach to it. However, this does not mean that you need to define all these potential relationships in your model. Neither does it mean that all these relationships have to be present in questions. For example, with give above, you can ask the question Who gives?

8.3 Specifying prepositional complements

Now select prepositions for the verb. Be careful to distinguish implicit prepositions, explicit prepositions, and verbs with particles.

8.3.1 Implicit prepositions

If the verb that you define will have any of the relationships described in "Defining relationships" on page 43, such as locative or temporal relationships, you need not specify a complement for these relationships; they are implicit and are automatically provided as a result of the relationship.

For example, if employees work in departments, do *not* specify the prepositional complement in for the verb *work*. Instead, link employee and department with one of the four *where* relationships. This gives the same coverage as explicit relationship plus *where* as a question word and relative pronoun.

For example, with the prepositions that are implicit with locative and temporal relationships, users can ask these questions:

*What departments do managers work in?
Which projects started before 1988?*

They can also ask questions involving the terms where and when, such as:

*List locations where employees work.
When did the interviews begin?*

8.3.2 Explicit prepositions

When these implicitly defined prepositions are not applicable, for example, when you do not need words like *when* and *where*, you need to explicitly specify the prepositions. For example, if you have created the verb *talk*, and you want the user to ask questions like *What did the customers talk about*, then you need to specify *about* as a preposition for the verb *talk*.

You select prepositions from the preposition complement panel. Another example is the verb *work*. If employees work on projects, you must specify the preposition *on* before defining the relationship between *work* and *projects*. When you have done this, the users can ask:

*What projects does John work on at HQ?
On which projects did John work?*

CUSTOMIZATION TOOL USER'S GUIDE

Define the verb *work* as shown in Figure 13 on page 69.

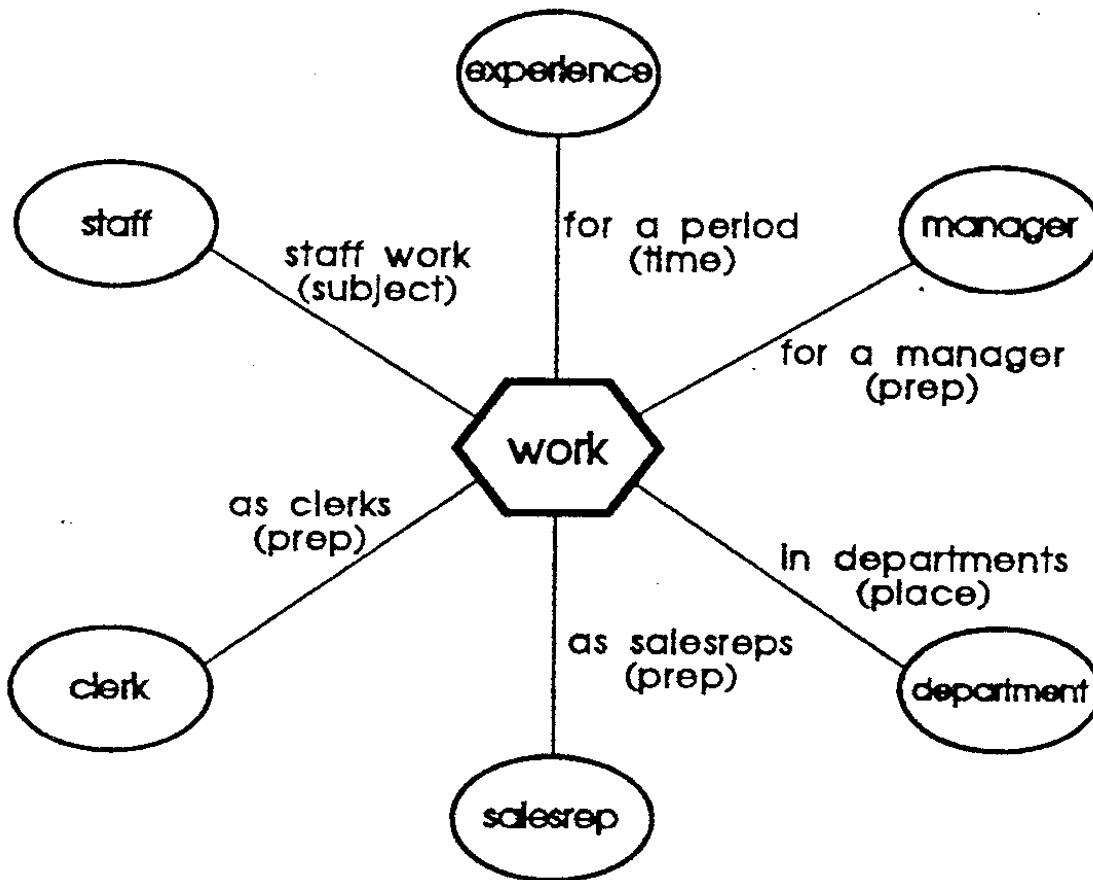


Figure 13 Verb *work* with prepositional complements and other relationships

The diagram shows that three prepositional relationships have been defined

Staff work as clerks
Staff work as salesreps
Staff work for a manager

8.3.3 Verbs with particles

Sometimes verbs have words associated with them that look like prepositions, but are actually inseparable parts of the verb. These parts of the verb are called particles.

For example, in: *John takes up his appointment on Monday*, *up* is a particle and not a preposition. You can tell the difference between a particle and a preposition by testing to see if you can ask questions with the particle/preposition at the beginning of the sentence.

The question makes sense:

On what does John work?

This does not:

CUSTOMIZATION TOOL USER'S GUIDE

Up what does John take?

In this second example, do not specify up as a preposition, but define the verb take up as one term.

Here are some other examples of verbs with particles:

What orders came in?

What companies were taken over?

What companies went under?

Who took on staff?

8.4 Defining verb relationships

After both verb and prepositional complements have been established, you define relationships between the verbs and their respective entities.

The main types of relationship that you define to verbs are those involving the subject and objects of the verb. These relationships act directly between the noun and the verb without the need for prepositions in questions. There are also two other relationships that do not involve prepositions: causal relationships and relationships of manner. When questions use prepositions, you must make a clear distinction between implicit and explicit prepositions, as with nouns and adjectives.

8.4.1 Subject/object relationships

Entities that are the subject or the object of the verb are linked without prepositions in the Customization Tool. However, you use the preposition *to* in questions involving an indirect object and the preposition *by*, when the verb is in the passive voice.

The four types of relationship that you can define to these verbs are:

- Nominative relationship (subject)
- Accusative relationship (direct object)
- Dative relationship (indirect object)
- Second accusative relationship (second direct object)

To define each of these relationships, select one of these from the list box of possible relationships between the noun and the verb:

Who interviews? manager (nominative)

What is bought? product (accusative)

To whom is someone/something given? book (dative)

What is someone/something named? module (second accusative)

You can form two other verb relationships that do not involve a preposition.

8.4.2 Causal relationships

If you have a column in a table that gives the reason for someone leaving your company, for example, you can define a verb leave and relate it to this column by clicking this line in the define relationships window:

CUSTOMIZATION TOOL USER'S GUIDE

Why does someone/something leave? reason for leaving

8.4.3 Relationships of manner

Some questions beginning with the word *how* are asking in what manner is something or other done. If you have a column with such a meaning, you can define a how relationship to it. Examples:

How was someone interviewed? interview method (by phone, in person) How did someone travel? means of transport

8.4.4 Implicit prepositions

The relationships to verbs that automatically let various prepositions be used in questions are locative and temporal duration relationships. They are defined in a similar manner to relationships between nouns, described in "Defining relationships" on page 43. Some examples of noun-verb relationships with implicit prepositions are:

Locative relationship: If you link an entity with a locative class to a verb (for example, *travel*, the entity can represent the answer to any of these questions:

*Where does someone travel?
From where does someone travel?
To where does someone travel?
Via which intermediate stages does someone travel?*

If you select one of these relationships, the same location words are available as for nouns. For a list of these words, and an example of how they are used, see "Location words" on page 46.

Time relationship: If you link an entity with a temporal class to a verb (for example, *travel*, the entity can represent the answer to any of these questions:

*When does someone travel?
Since when does someone travel?
Till when does someone travel?*

Duration relationship: If you link an entity with a duration class to a verb (for example, *travel*, the entity can represent the answer to this question:

How long does someone travel?

If you select one of these relationships, the same time words are available as for nouns. For a list of these words, and an example of how they are used, see "Time words" on page 48.

8.4.5 Explicit prepositions

Explicit prepositions between nouns and verbs are defined in a similar manner to noun-noun relationships involving prepositions, described in "Prepositions between nouns" on page 51. For example:

CUSTOMIZATION TOOL USER'S GUIDE

What does employee work as? clerk
What does employee work on? project

8.5 Verbs in passive voice

Sometimes users ask questions in the passive rather than the active voice, using the preposition *by*. Do not be misled by these types of questions into thinking that you need a prepositional relationship using *by*.

A verb is in the active voice if the subject is the agent of the action, and in the passive voice if the subject is not the agent of the action, or it does not have a subject at all. For example:

Who interviews applicants?

Managers interview applicants (interview in the active voice).

Who were interviewed by the managers?

The applicants were interviewed by the managers (interview in the passive voice).

In this example, define interview in the normal way. The preposition *by* is then automatically available to users in the passive sense.

If the users want to use *by* to refer to a place, *by* is available with the *via* locative relationship.

Some verbs are only generally used in the passive voice, but without the subject of the verb. The verb locate in EPEDEMOE is an example. Such verbs can be defined with any of the verb forms that use a direct object, but without a subject relationship being defined. Users could then ask questions such as

What department is located in Chicago?

To define located as an adjective is an alternative that is described in "Creating adjective subclasses" on page 55.

The verb *to be born* should not be defined with *be* being conjugated.

8.6 Entities referring to numerical data

"Measured-by relationship." on page 49, described how to classify column entities -that contain numerical data values and the measured-by relationships that link to them. However, some situations involving numerical data involve creating new entities. This section covers when these are needed.

8.6.1 Counted-by relationships

Normally, when users ask a question beginning with *How many*, Ergo will provide a count of the records that meet the particular criterion specified in the rest of the question. However, sometimes the answer to a *How many* question is contained in the data values themselves.

CUSTOMIZATION TOOL USER'S GUIDE

For example, a table might contain a column denoting a count of items in an order or the number of inhabitants in a country. But if you customized the column as item or inhabitant, and asked *How many items/inhabitants are there*, you get a count of the number of records containing items/inhabitants.

To let the natural language engine distinguish between these two situations-when to count the records and when to use the data values in the database - you must define a new entity that can be used in *How many* questions. This new entity gives a second view of columns classified as quantity.

For example, in EPEDEMOE, the order quantity column of the orders table can be seen as either a magnitude or a count of items in each order. Similarly, in a countries table, a column containing information about how many people live in the countries could be interpreted either as population or the number of inhabitants.

When a quantity is viewed as a count, it looks like a total or sum of individual items. For example, in a products database, there might be one row in the table for each product, which would then have an individual serial number, but in many situations, individual products are not recorded individually; a count is maintained of the quantity in stock, for example. Similarly, a country might maintain a database of all its citizens, where information about each person is contained in one record per person. However, in a countries database containing information about countries as a whole, information about inhabitants is only recorded in the aggregate.

To customize the two ways of referring to quantities:

- Classify the column entity as a quantity and link it to the table entity with a possessive or measured-by relationship.
- Or:
 1. Create a new entity called, for example, item or inhabitant, classified in whatever way might be suitable. There is no need to define an SQL statement for this entity. This represents the count view of the quantity.
 2. Link this new entity to the quantity by selecting the counted-by relationship in the define relationship window, like this:

How many items are there? magnitude
How many inhabitants are there? population
 3. Create an appropriate relationship between the new entity and the table entity. For example, a locative relationship for the countries and a possessive relationship for the orders.

With these customizations, users can ask:

How many items per order are there?
How many items for order 3456 are there? ‘
How many items does each order have?
How many items does (each) order 3456 have?
How many inhabitants does Sweden have?
How many inhabitants are there in Sweden?

However, if users want to ask questions like this:

How many items are there per order?
How many items are there for order 3456?

CUSTOMIZATION TOOL USER'S GUIDE

you must add prepositions to the noun item and create an explicit prepositional relationship with order. This is because the natural language engine cannot generate the implicit prepositions from the possessive relationship in this case. Note that if users ask questions such as *How many items per order are there?* they will receive two interpretations.

8.6.2 Unit entities

To give maximum language coverage, an entity classified as a quantified property must have a corresponding unit of measure associated with it. For example, if you classify an entity as *value*, you could choose a unit of measure *dollar* corresponding to this.

Ergo provides many units of measure that you can choose from. Table 2 on page 42 shows the relationships between quantified properties, units of measure, and the units they are measured in.

However, if you have a column that contains values in different units from those provided, then you must create a new entity for that unit. For example, to create a unit of currency for the Dutch guilder, create a new entity with a term *guilder* and make it an instance of *currency*. When you now classify an entity as either a price or a value and select unit of measure, you will find that *guilder* has been added to the units available.

You can create new units that are instances of the *unit-of-measure* class itself rather than one of its subclasses. Here, the unit can only be used with an entity classified as *size*.

8.6.3 Quantified_property and unit_of_measure entities

If you want to classify an entity as a quantified-property, but with a unit of measure that is not included in Table 2 on page 42, then you need to create two new entities plus entities for as many units that you want to measure the quantified-property in. You define these entities as subclasses of *quantified-property* and *unit-of-measure*.

For example, if you have a column containing information about temperature in Fahrenheit, follow these steps:

1. Create an entity for a new subclass of *quantified-property* called *temperature*.
2. Create an entity for a new subclass of *unit_of_measure* called *unit_of_measure_temp*. There is no need to define a term.
3. Create a unit-type relationship between these two entities.
4. Create an entity as an instance of *unit_of_measure_temp* called *Fahrenheit*.
5. Define a term, *Fahrenheit*, for the entity.
6. Classify the column entity as a subclass of *temperature* with a unit of measure *Fahrenheit*.

CUSTOMIZATION TOOL USER'S GUIDE

9 TESTING THE CONCEPTUAL MODEL

This chapter describes how you test your conceptual model with the questions that the model is designed to answer. How often you need to do this will depend on your experience and the complexity of the model.

Initially, when you are learning to customize, it is a good idea to test the model whenever you have customized a new type of entity. For example, after defining the basic table and column entities or a few verb entities. You should also test the model thoroughly before making it available to users.

To test your conceptual model, you use one of the query interfaces available with Ergo to ask questions.

9.1 Making the conceptual model available

The format of the conceptual model is different in the Customization Tool from that used by the natural language engine. So before you can test your model, you convert the model and make it available to the engine.

In the query interface, the conceptual model is called a language model, or model for short. After starting the particular query interface that you are using, you must specify the name of the model that you want to test if the name has changed.

9.2 Asking questions

When you have uploaded the conceptual model, you can use the query interface to ask questions.

When you ask a test question, the query interface shows you:

- A natural language interpretation (or several possible interpretations) of the question
- One or more SQL statements for each interpretation .The answer to the interpretation you choose

Check that:

- An interpretation has the same meaning as the natural language question.
- The SQL statement matches the interpretation.
- The answer is correct.

9.3 Questions that use only nouns

If you have defined nouns but have not yet defined verbs, you can ask some simple questions:

- Questions with command words:

List staff.
Show employees.

- Questions with forms of have (if you defined possessive relationships):

Who has the highest salary?
Does Sanders have any commission?

- Questions with forms of be:

CUSTOMIZATION TOOL USER'S GUIDE

*How many employees are there?
Who is manager of department 20?*

9.4 Questions that use verbs

When you define verbs, you can ask a wider variety of questions, such as:

*Which department does Molinare manage?
How many projects are run by Pacific department?
What interviews did Sanders conduct?
Who manages the department that runs the project that makes bushing? How many employees work for Plotz?*

9.5 Questions that use adjectives

When you define adjectives, you can ask questions such as:

*Is Plotz a senior manager?
Which orders for Rothman were valuable?*

9.6 Getting information about your model

There are a number of ways to get information about your model. One of these is to run the model report utility described in Chapter 15, "Producing reports" on page 134. You can also obtain information about the model when using the query interfaces. This is most useful in finding out why a question does not work in the way that you think it should.

9.7 Dealing with problems

This section lists some problems that you may encounter when testing your conceptual model, and possible solutions:

- Ergo cannot interpret your question.
In most cases, you should get an error message that indicates the reason for the problem. Consult the online help for the message for advice on how to handle the problem.
- The query interface returns your question with some words highlighted.

Other than misspellings, each highlighted word is either a data value or a term that you have not defined in the conceptual model. If the highlighted word is not a data value, add the term to the conceptual model, or rephrase the question to avoid using the term.

- No interpretation matches your question.
For each entity referred to in the question, check the terms, classes, and relationships.
- Many interpretations are produced.
You may have:
 - Defined unnecessary relationships between entities.
 - Classified an entity incorrectly.
 - Used the same term for more than one entity. (This is not necessarily an error, but it can produce many interpretations.)

CUSTOMIZATION TOOL USER'S GUIDE

- The SQL statement does not match your question.
For each entity referred to in the question, check the terms, classes, and relationships.
- The SQL statement is correct, but the answer is not.
Check that the SQL statement exactly matches the data value in the database. For example, the SQL statement may refer to MOTORS while the database contains the value MOTOR. In this case, you should define an entity for motor, and classify it as a subclass or instance of product.
- An expression returns an unexpected null value.
In SQL, any expression that involves a null value will return a null value. To avoid this problem, see "Handling null values" on page 61.
- A question that requires *temporary tables* is not executed.
For some questions, including questions with a range (such as from A to B) or ranking (such as *the three largest*), Ergo must create a temporary table. You may not have the authority or enough disk space to create the temporary table.

9.8 Making the model available to users

When you have finished testing the model you follow the instructions on dialoguetech.com in order to embed the model in an application.

Part 3. The graphical interface

10 MANAGING THE MODEL

This chapter tells you how to operate the graphical interface of the Customization Tool except for specifying entities and relationships. These functions are described in Chapter 11, "Specifying entities" on page 104 and Chapter 12, "Specifying relationships" on page 130, respectively. For further details, see the online help.

You choose options from the action bar or manipulate the diagram directly to manage your conceptual models. These items from the action bar are available:

<i>File</i>	To print the diagram, to save the conceptual model, or to specify file properties.
<i>Edit</i>	To copy, move, and delete icons in the diagram and to search for icons.
<i>Action</i>	To create new entities or group entity icons in a cluster.
<i>Mode</i>	To switch between database and language mode.
<i>View</i>	To scale and position the diagram and to show name labels.
<i>Options</i>	To change the colors and patterns of the icons.
<i>Help</i>	To get help on any field or selectable item in a window.

10.1 File

Use these *File* actions to work with conceptual models:

New

Removes the open conceptual model and lets you start a new one.

Open

Displays and lets you edit an existing conceptual model.

Properties

Displays and lets you change the properties of the open conceptual model.

Save

Saves the open conceptual model under its current name.

Save as

Saves the open conceptual model under a new name.

Print

Produces a printed copy of the displayed diagram.

CUSTOMIZATION TOOL USER'S GUIDE

Exit

Exits from the Customization Tool.

10.2 New

Use *New* to start a new conceptual model.

Because you can work on only one conceptual model at a time, the open conceptual model is closed. If you have changed the conceptual model since the last time you saved it, you get a chance to save your changes. The new conceptual model remains untitled until you save it.

10.3 Open

Use *Open* to open an existing conceptual model.

Because you can work on only one conceptual model at a time, the open conceptual model is closed. If you have changed the conceptual model since the last time you saved it, you get a chance to save your changes.

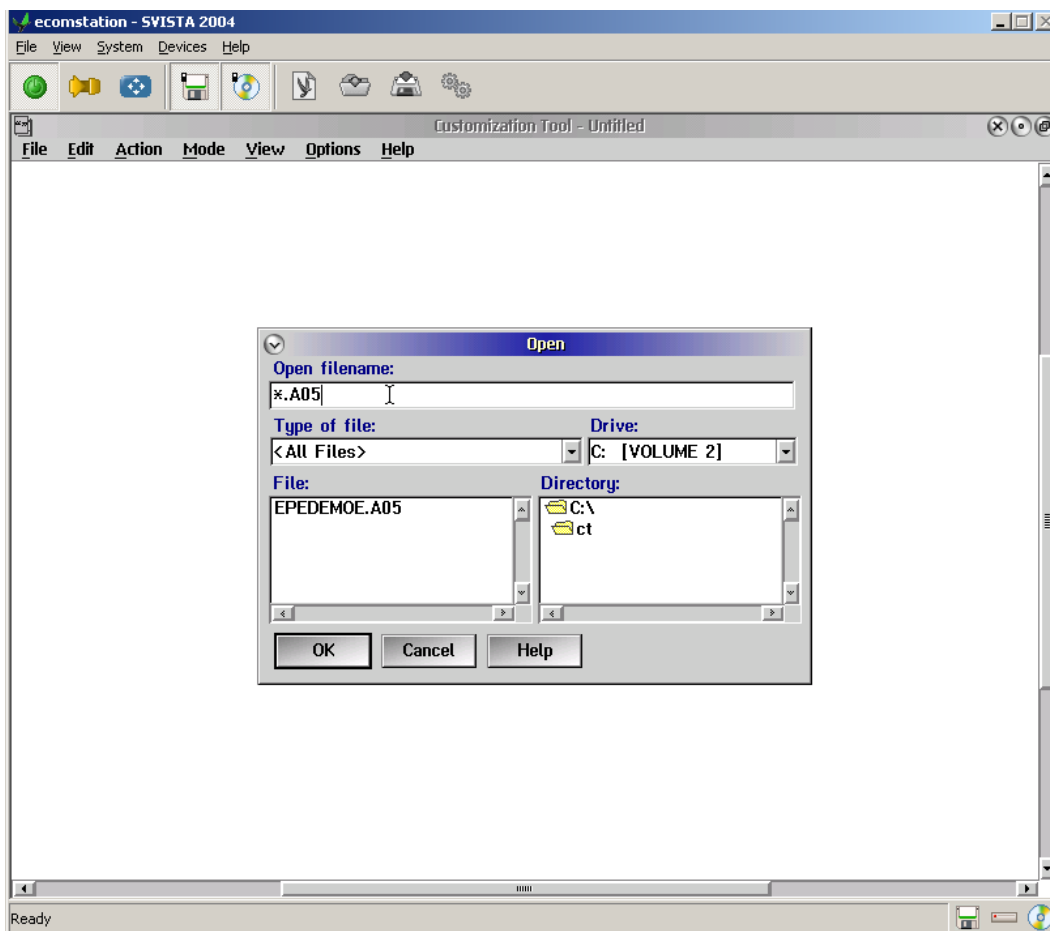


Figure 14. Open model.

10.4 Properties

CUSTOMIZATION TOOL USER'S GUIDE

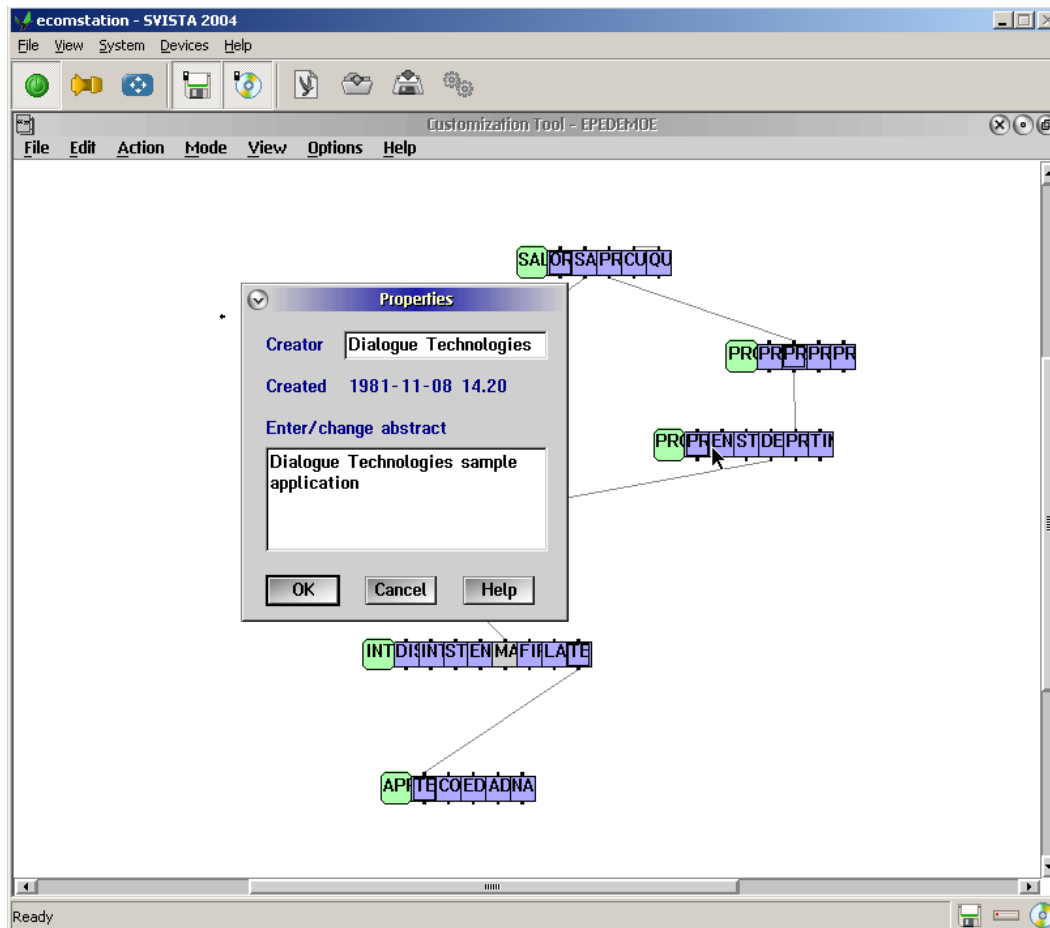


Figure 15. Define model properties.

To enter or change a description of the conceptual model, select *Properties* from *File*. In the window that appears, type:

- Your name in the *Creator* field
- A brief description of the conceptual model in the *Abstract* field

When the conceptual model is made available to users they can ask questions about who created it and what it contains.

10.5 Save

Use *Save* to save the open conceptual model under its current name. If the conceptual model is untitled (new), you must enter a name. After saving the conceptual model, you can continue working with it.

10.6 Save as

Use *Save as* to save the open conceptual model under a different name. After saving the conceptual model, you can continue working with it.

CUSTOMIZATION TOOL USER'S GUIDE

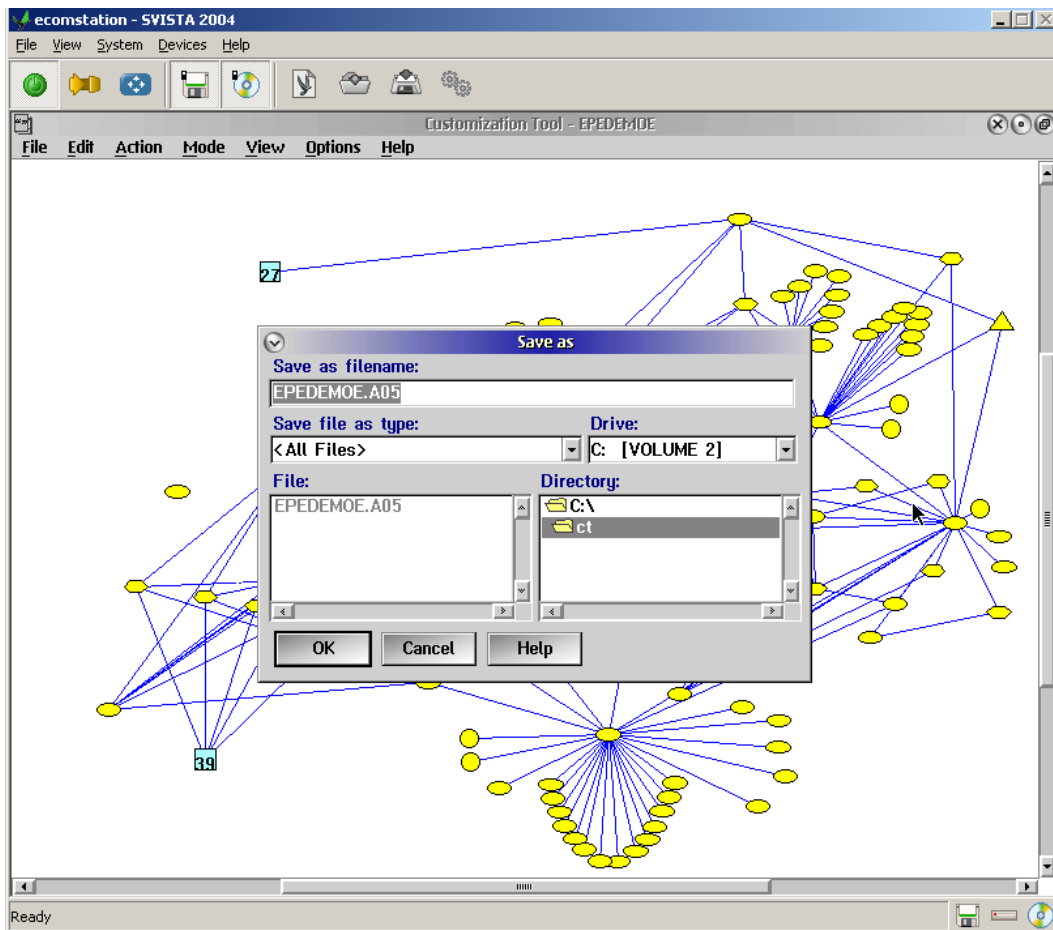


Figure 16. Save model with new name or directory.

10.7 Print

Use *Print* to send a copy of the diagram to your local printer.

If the screen contains only a portion of the diagram, only the portion contained on the screen is printed.

To see (and print) more of the diagram, select *Scale diagram* from *View*. By decreasing the size of the sample icon, you can fit more icons into the screen area.

10.8 Exit

Use *Exit* to exit the Customization Tool.

You can also exit the Customization Tool in these ways:

- Select *Close* from the system icon.
- Double click on the system icon.

10.9 Edit

Use these *Edit* items to edit the diagram:

CUSTOMIZATION TOOL USER'S GUIDE

Undo

Reverses the effect of the last movement action.

Copy

Copies the selected entity, group of entities, or cluster to the clipboard.

Paste

Pastes the contents of the clipboard to the selection point.

Clear

Erases the selected table, entity, entities, or cluster from the conceptual model.

Move

Moves the selected icon to the selected position.

Find

Searches for an icon in the diagram.

10.10 Undo

Use *Undo* to reverse the effect of one of these actions:

- *Center selected* from *View*
- *Fit diagram* from *View*
- *Move* from *Action*
- Dragging an icon with the mouse

You must select *Undo* immediately, before selecting any other action.

10.11 Copy

Use *Copy* to copy the selected entity, entities, or cluster to the clipboard. The relationships between the entities are also copied except for undefined, initial relationships. Relationships to entities or clusters outside the group being copied are not copied.

You can use *Copy* to copy entities and clusters within a model or to another model.

10.12 Paste

Use *Paste* to paste the contents of the clipboard to the selection point. This option is not available if an entity, group of entities, or a cluster is selected.

10.13 Clear

In database mode, use *Clear* to erase the selected table from the diagram.

If there are dependencies between the selected table and other tables, these dependencies are also erased.

In language mode, use *Clear* to delete the selected entities and clusters from the conceptual model.

CUSTOMIZATION TOOL USER'S GUIDE

You can delete individual entities, groups of entities selected within a frame, or entities within a cluster. However, if a frame or cluster contains clusters, these are not deleted. To remove the entities within a cluster, the individual cluster must be specifically selected.

If there are relationships between the selected entities and other entities, these relationships are also deleted.

10.14 Move

Use *Move* to move an individual entity, cluster, table, or column icon or a group of icons. Unlike *Copy*, *Move* does not use the clipboard. Its purpose is solely to help you arrange the icons in your diagram.

To move an individual icon using a mouse:

1. Hold down the right mouse button and drag the icon to where you want to move it.
2. Release the mouse button.

To move a group of icons using a mouse:

1. Position the pointer at one corner of the area.
2. Hold down the left mouse button and drag the pointer to the opposite corner of the area.
3. Release the mouse button. A dotted frame surrounds the area you have selected.
4. Point the arrow cursor inside the frame (not on an icon). Hold down the right mouse button and drag the frame to where you want to move it.
5. Release the mouse button.

Note: You can also move individual icons into and out of frames marking groups of icons using the right mouse button. Do not press the left mouse button because this will change the selection of the icons

To move an individual icon using the keyboard, without changing the selected icon(s):

1. Use the arrow keys to move the pointer inside an icon.
2. Hold down the *Shift* key and press the *spacebar* to mark the icon as movable.
3. Use the arrow keys to move the pointer to the new location for the center of the icon.
4. Hold down the *Shift* key and press the *spacebar*.

To move an individual icon using the keyboard, changing the selected icon:

1. Use the arrow keys to move the pointer inside an icon.
2. Press the *spacebar* to select the icon.
3. Use the arrow keys to move the pointer to the new location for the center of the icon.
4. Select *Move* from Action with the keyboard. To do this:
 - a. Press *F10*.
 - b. Move to the option with the arrow keys.
 - c. Press *Enter*

Alternatively, hold down the *Ctrl* key and press *m*.

Do not use the mouse to invoke the action bar.

To move a group of icons using the keyboard:

1. Use the arrow keys to place the pointer at one corner of the area.

CUSTOMIZATION TOOL USER'S GUIDE

2. Hold down the *Ctrl* key and press the *spacebar*.
3. Use the arrow keys to move the pointer to the opposite corner of the area.
4. Hold down the *Ctrl* key and press the *spacebar*. A dotted frame surrounds the area that you have selected.
5. Use the arrow keys to move the pointer to the new location for the center of the frame.
6. Select *Move* from *Action* with the keyboard, or hold down the *Ctrl* key and press *M*. Do not use the mouse to invoke the action bar.

Note: You can also move individual icons into and out of frames using the keyboard only. To do this, use the keyboard method for individual icons, without changing the selected icon(s), described above.

10.15 Find

Use Find to find an icon in the diagram. The window in Figure 17 is for language mode. There is a similar window for database mode.

If you have a large diagram with many icons, this action may save you time.

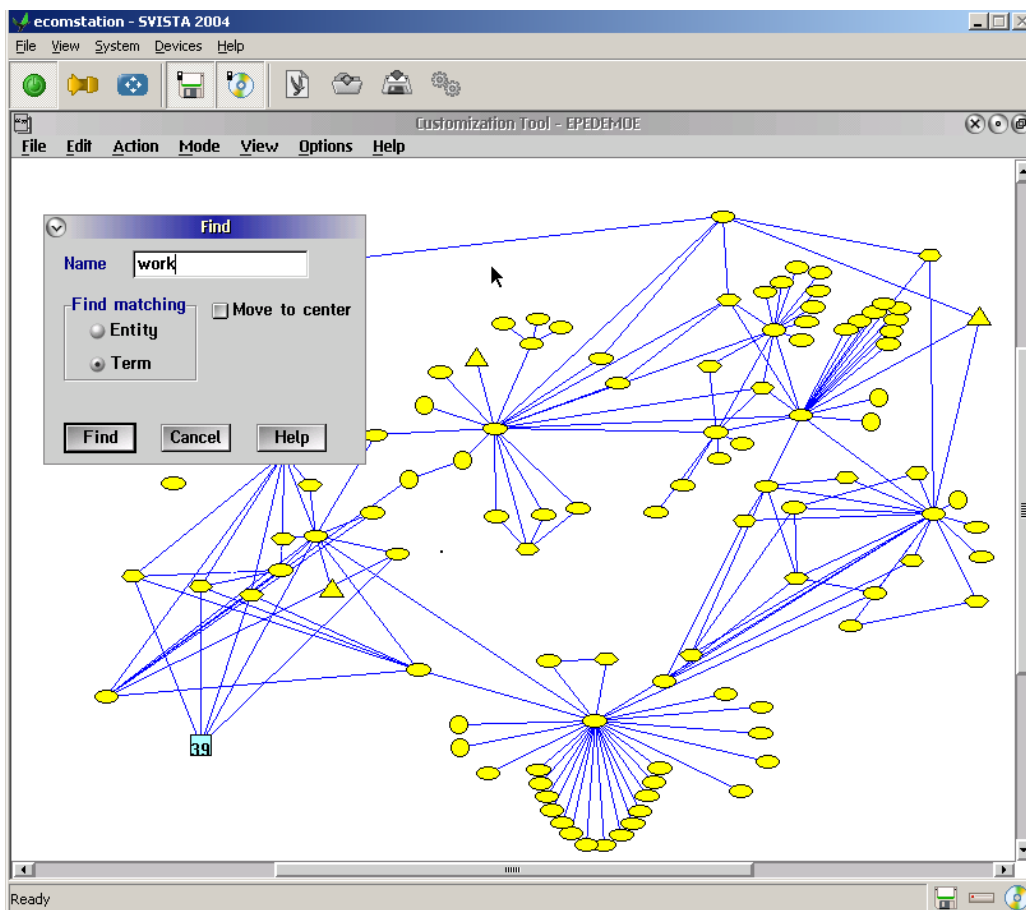


Figure 17. Find term or entity in language mode.

10.16 Action

Use these *Action* items to manage icons in the conceptual model:

CUSTOMIZATION TOOL USER'S GUIDE

Add tables

Adds tables from the database to your conceptual model.

Collapse table

Hides the columns of the selected table.

Hide columns

Lets you hide columns from the selected table.

Create entity

Creates a new entity at the selected position.

Cluster

Groups the selected entities into a cluster.

Expand

Shows the constituents of the selected icon.

Define

Lets you work with the selected icon.

Rotate

Rotates group of icons by 90 degrees.

Snap to grid

Snaps group of icons to grid.

10.17 Add tables

Use Add tables to add tables to your conceptual model in database mode.

You must have previously extracted database catalog information, as described in Chapter 13, "Extracting database catalog information" on page 133.

If the structure of a table changes (for example, a new column is added), extract database catalog information again, and add the table again. The previous version of the table is removed from the diagram, and a new version (with the keys and dependencies from the original table) is added.

See "Adding tables to the conceptual model" on page 23 for further information on adding tables.

10.18 Collapse table

Use Collapse table to hide the columns of the selected table. Only the table icon remains visible.

The two ways to display the columns again are:

- Select the table icon and then select *Expand* from *Action*.
- Double click on the table icon.

CUSTOMIZATION TOOL USER'S GUIDE

10.19 Hide columns

Use *Hide columns* to hide and exclude columns in database mode.

Hide columns in a table when you do not want to use them in the conceptual model. They are hidden from view in both database mode and language mode.

Exclude columns in a table when you do not want to work with them in language mode, but they are needed in database mode. This most often happens when the two ends of a join path represent the same concept. In this case, exclude one of the columns so that only one is displayed in language mode.

You can also exclude individual columns when defining column information. This is described in "Excluding columns" on page 29.

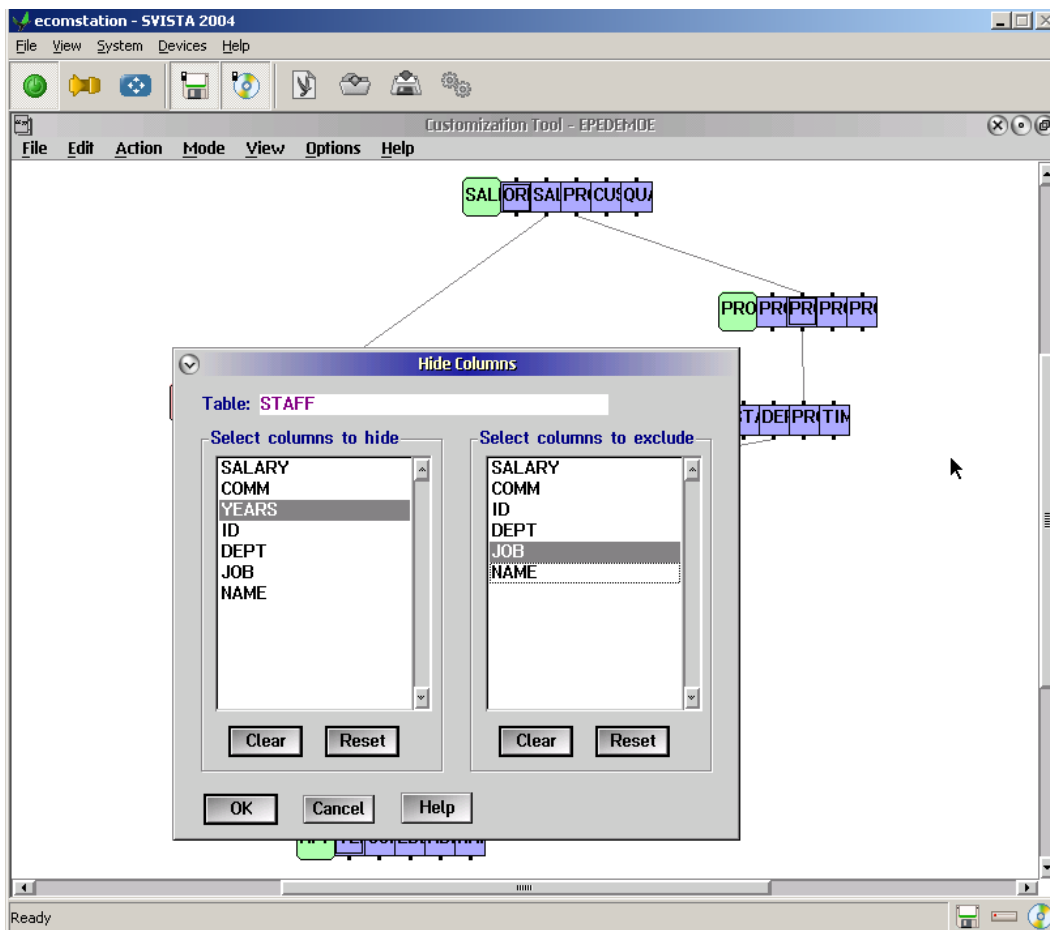


Figure 18. Hide columns in database mode.

10.20 Create entity

Use *Create entity* to add a new entity to your conceptual model at the selected position.

To select a position for the entity icon:

1. Place the pointer where you want the new entity.
2. Click the left mouse button (or press the *spacebar*) to position the diamond-shaped marker.

CUSTOMIZATION TOOL USER'S GUIDE

3. Select *Create entity*.

See Chapter 11, "Specifying entities" on page 104 for details on creating entities.

10.21 Cluster

Use *Cluster* to replace the selected group of icons with a cluster icon. This reduces the number of icons that are displayed in the diagram. For example, you can group entities that logically belong together, such as a table entity with its column entities. You can include entity icons, other cluster icons, or both.

You can create a cluster with a mouse or with the keyboard.

With a mouse:

1. Position the pointer at one corner of the area.
2. Hold down the left mouse button and drag the pointer to the opposite corner of the area.
3. Release the mouse button. A dotted frame surrounds the area that you have selected.
4. Select *Cluster* from *Action*.
5. Change or verify the suggested cluster name.

With the keyboard:

1. Use the arrow keys to place the pointer at one corner of the area.
2. Hold down the *Ctrl* key and press the *spacebar*.
3. Use the arrow keys to move the pointer to the opposite corner of the area.
4. Hold down the *Ctrl* key and press the *spacebar*. A dotted frame surrounds the area that you have selected.
5. Select *Cluster* from *Action*.
6. Change or verify the suggested cluster name.

CUSTOMIZATION TOOL USER'S GUIDE

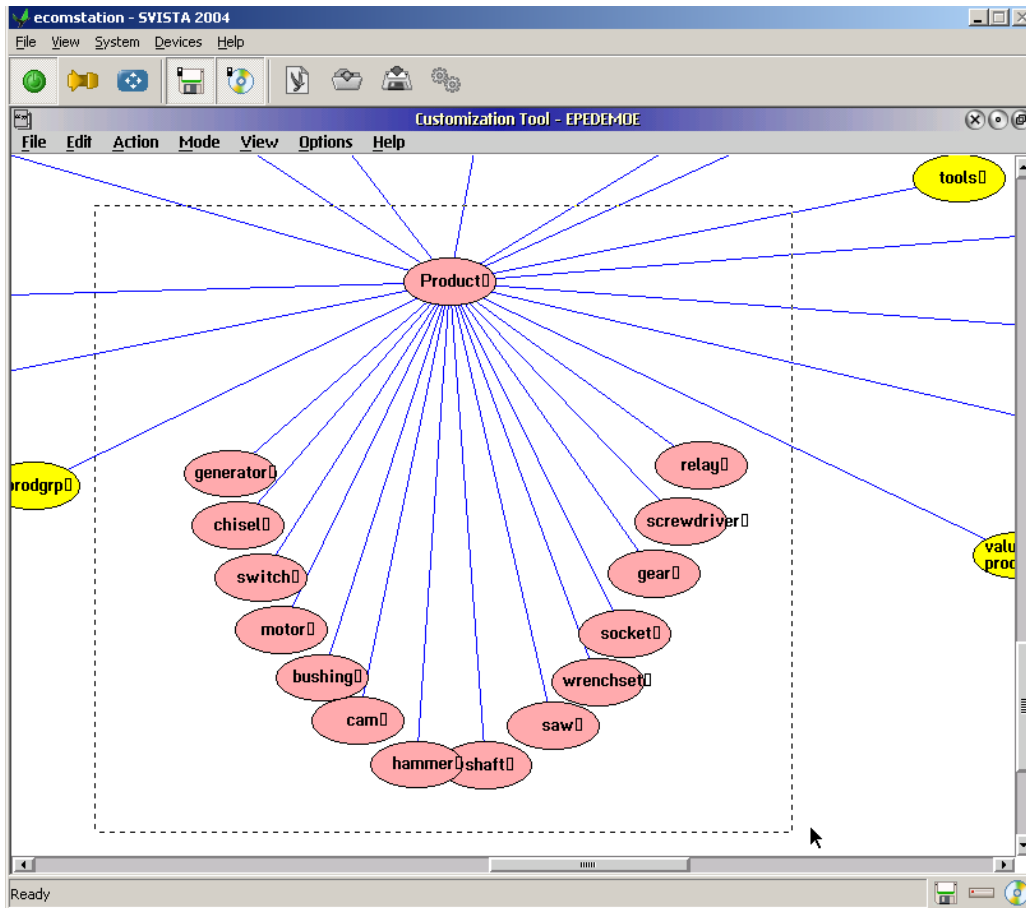


Figure 19. Creating a cluster.

A cluster is represented by a square icon in the diagram.

There are two ways to display the entity icons again:

- Select the cluster icon and then select *Expand* from *Action*.
- Double click on the cluster icon.

CUSTOMIZATION TOOL USER'S GUIDE

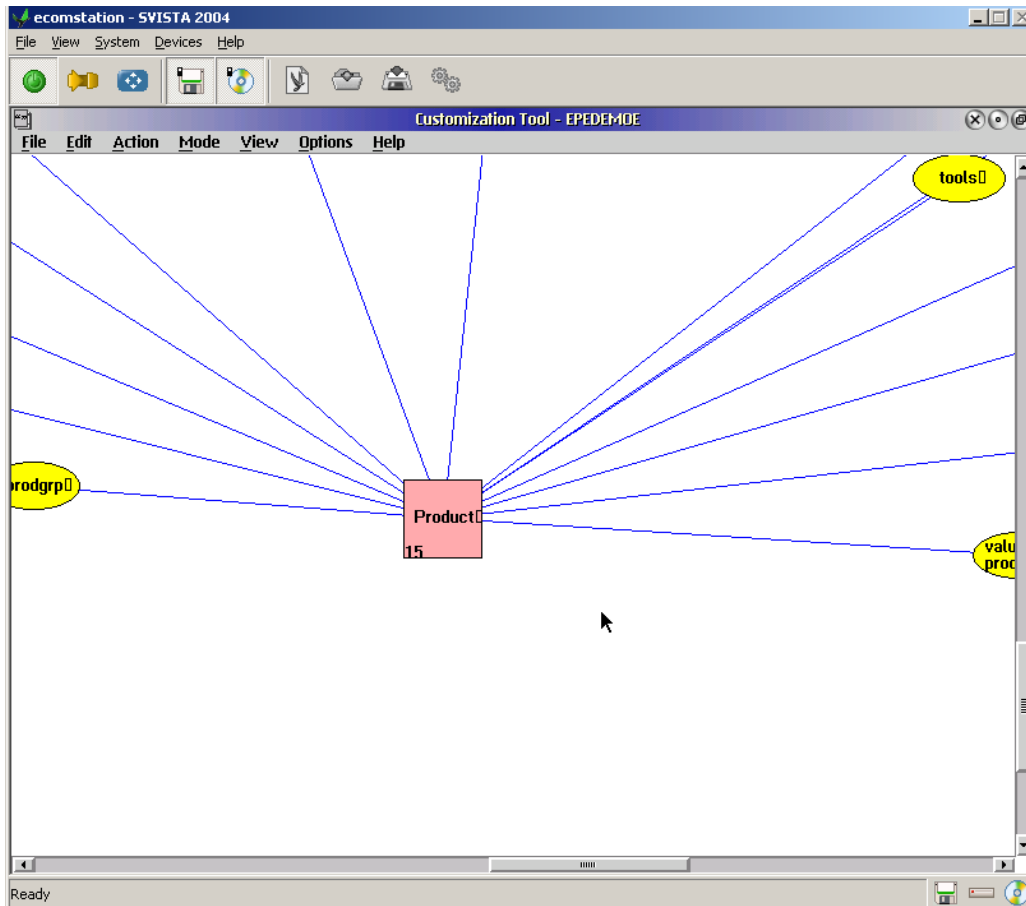


Figure 20. Language diagram with cluster.

10.22 Expand

Use *Expand* to display the columns of the selected table icon or the entities of the selected cluster icon.

You can also expand a table or cluster icon by double clicking on it with the left mouse button.

10.23 Define

Use *Define* to work with the selected entity, relationship, column, or dependency icon, the entities within a selected cluster, or the entities in a selected group of icons.

If a cluster or a group of icons is selected, you can define all these at the same time, using suggestions made by the Customization Tool as described in "Handling multiple entities" on page 127.

For further information on defining individual entities, relationships, columns, and dependencies, see:

- Chapter 11, "Specifying entities" on page 104
- Chapter 12, "Specifying relationships" on page 130
- "Defining column information" on page 27
- "Defining dependencies" on page 29

CUSTOMIZATION TOOL USER'S GUIDE

10.24 Rotate

Use *Rotate* to rotate the selected group of icons through 90 degrees, counter clockwise around the frame center. The positions of the icons are rotated, not the icons themselves.

This is useful when switching from database mode to language mode.

10.25 Snap to grid

Use *Snap to grid* to center all icons in a selected frame on a mesh point in the grid. This is useful after you move a frame, because the icons are not snapped to grid in order to maintain their relative position to each other. Selecting Snap to grid then provides you with a way of aligning all the icons in the frame onto the grid.

If you select a single entity or cluster, this action is equivalent to clicking the right mouse button.

10.26 Mode

Use these *Mode* actions to switch between the two customization diagrams:

Database

Lets you work with tables and columns.

Language

Lets you work with entities and relationships.

10.27 Database

Use *Database* to switch from language mode to database mode.

Before you use database mode for the first time, download table information from the database, as described in Chapter 13, "Extracting database catalog information" on page 133.

In database mode:

1. Select the tables that you want to add to the conceptual model.
2. Specify column information.
3. Define dependencies between columns.

When you finish working with tables, columns, and dependencies, select *Language*. Tables and columns are replaced by entities. Proposed relationships are also shown.

10.28 Language

Use *Language* to switch from database mode to language mode.

The tables and columns are replaced by entities. Proposed relationships are also shown.

CUSTOMIZATION TOOL USER'S GUIDE

In language mode, you can define entities and the relationships between them.

To change the appearance of the diagram, use the *View* actions. To add more tables or change column definitions, return to database mode.

10.29 View

Use these *View* actions to change the display of the diagram:

Scale diagram

Changes the size of the diagram.

Fit diagram

Changes the size and position of the diagram, so that all icons fit in the window.

Overview

Provides an overview of the complete model.

Center selected

Moves the diagram, so that the selected icon or position appears in the middle of the window.

Refresh

Draws the diagram again.

10.30 Scale diagram

Use *Scale diagram* to change the size of the diagram.

You can also expand the diagram by pressing + on the keyboard, or contract the diagram by pressing - on the keyboard.

The currently selected icon (or position on the screen) does not move. All other icons move closer to the selected position (if you contract the diagram), or farther from the selected position (if you expand the diagram).

CUSTOMIZATION TOOL USER'S GUIDE

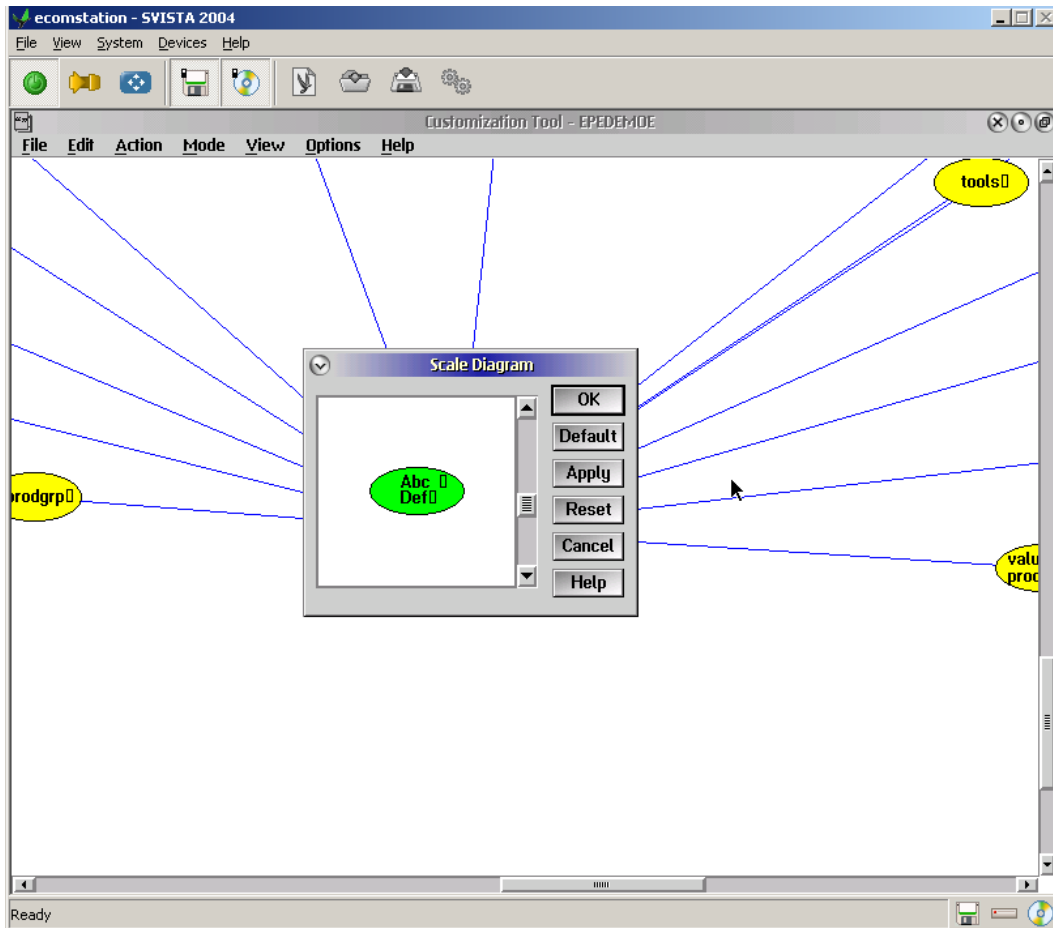


Figure 21. Scaling the diagram.

10.31 Fit diagram

Use Fit diagram to change the size and position of the diagram so that all icons fit within the window.

10.32 Overview

Use Overview to display a miniature of the complete diagram, including parts that are outside the main window. The border of the main window is shown in the overview window as a dotted rectangle.

CUSTOMIZATION TOOL USER'S GUIDE

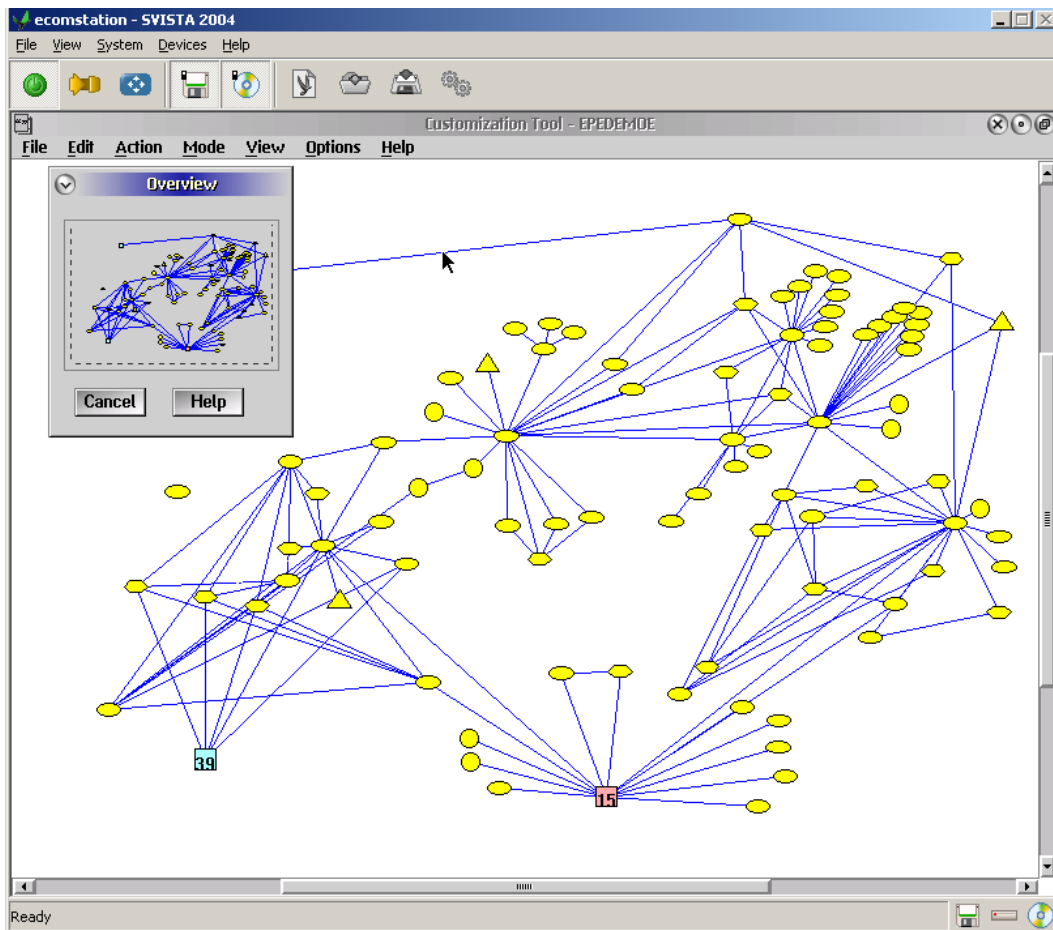


Figure 22. Overview of the whole diagram.

You can see another part of the diagram in the main window without changing the scale by clicking the left mouse button in the overview window where you want the new center of the diagram to be.

You can change the part of the diagram that you want to view in the main window, possibly changing the scale. To do this, press the left mouse button and drag across the overview window. The Customization Tool draws a dotted rectangle indicating the border of the main window. The main window is redrawn when you release the mouse button.

After scaling the main diagram or changing the size of the window, you can update the overview window by clicking in its title bar.

10.33 Center diagram

Use *Center diagram* to move the diagram so that the selected icon or position appears in the middle of the window:

1. Select an icon or position on the screen.
2. Select *Center diagram*.

10.34 Refresh

Use *Refresh* to draw a fresh diagram.

CUSTOMIZATION TOOL USER'S GUIDE

10.35 Options

Use these Options actions to customize your use of the Customization Tool.

Styles

Lets you change the colors and patterns of icons.

Show labels

Displays the name of an icon when you select it.

Show information area

Displays the information area in the *Define Entities* window.

Grid

Lets you select a visible grid, an invisible grid, or no grid.

Name break

Lets you select icon name-break characters.

Preferences

Displays and lets you change user preferences.

10.36 Styles

Use these *Styles* items to change the colors and patterns of icons in database mode:

CUSTOMIZATION TOOL USER'S GUIDE

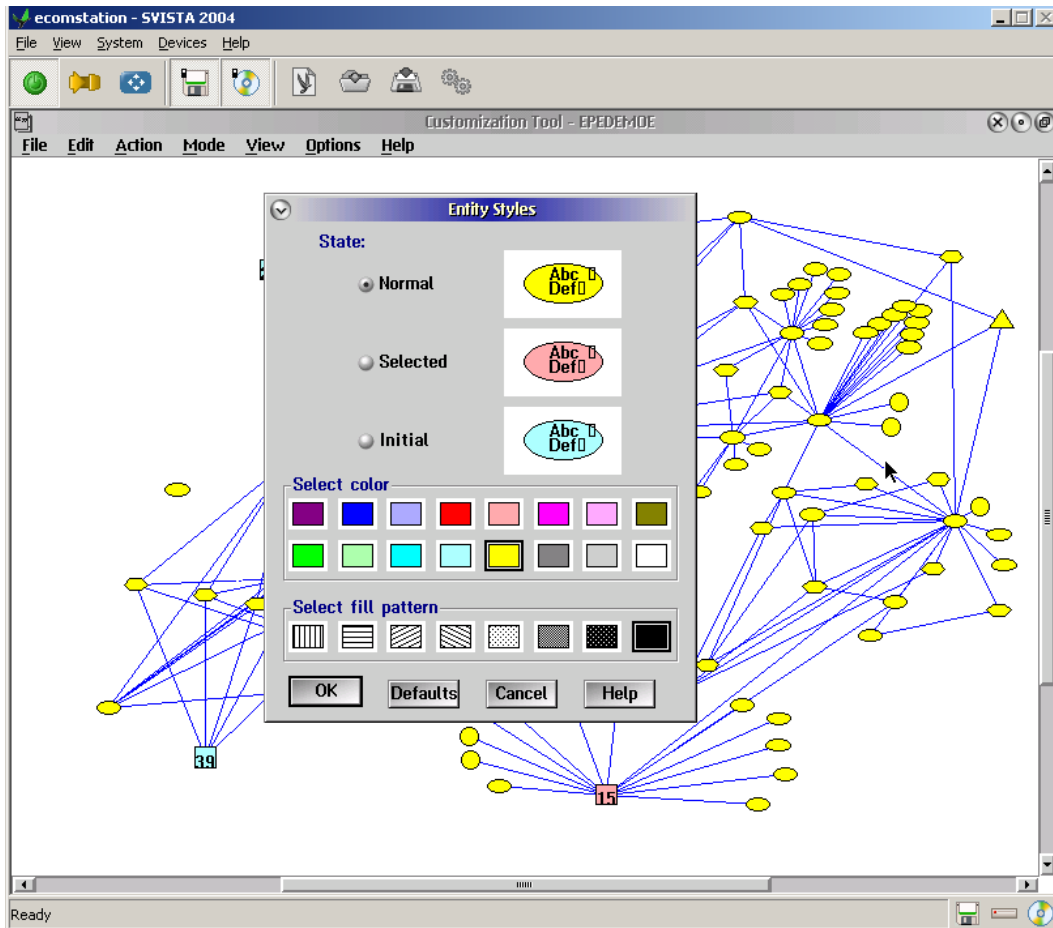


Figure 23. Defining styles of entity icons.

Table
Column
Dependency

Use these *Styles* items to change the colors and patterns of icons in language mode:

Entity
Relationship
Cluster

CUSTOMIZATION TOOL USER'S GUIDE

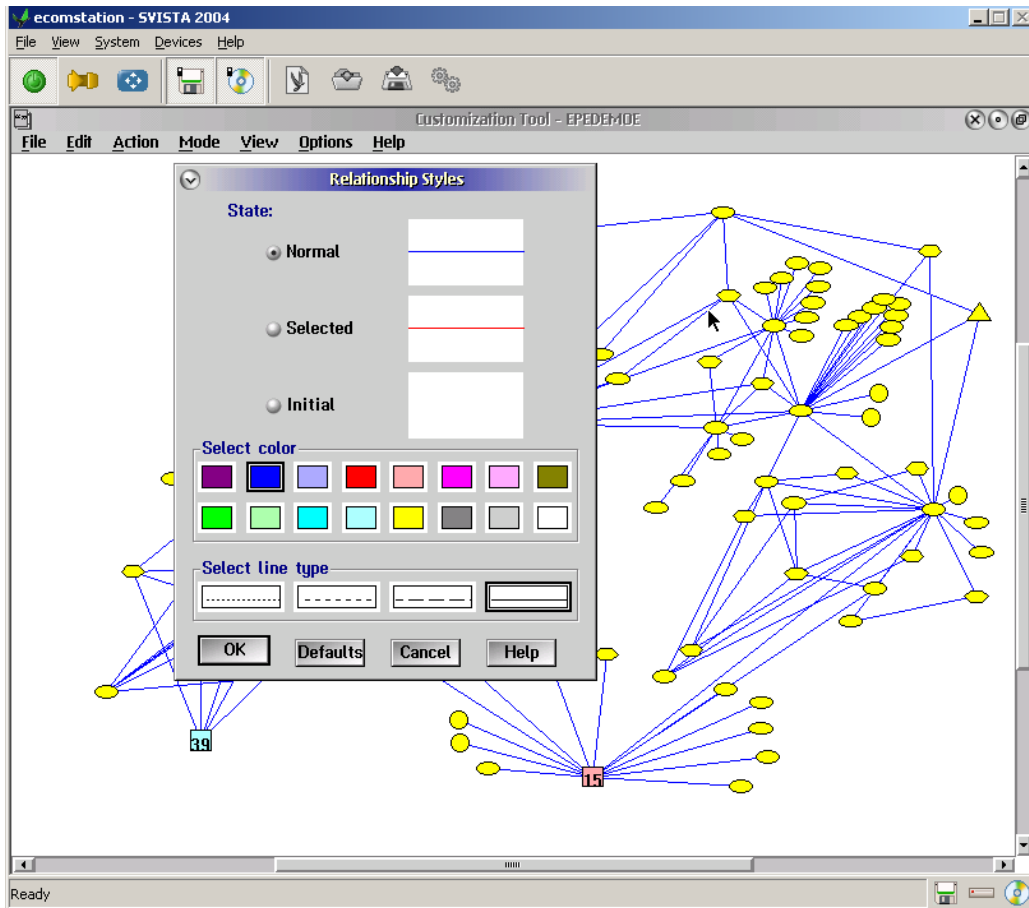


Figure 24. Defining styles of relationship connectors.

Use *Others* to change the colors and patterns of all other objects.

CUSTOMIZATION TOOL USER'S GUIDE

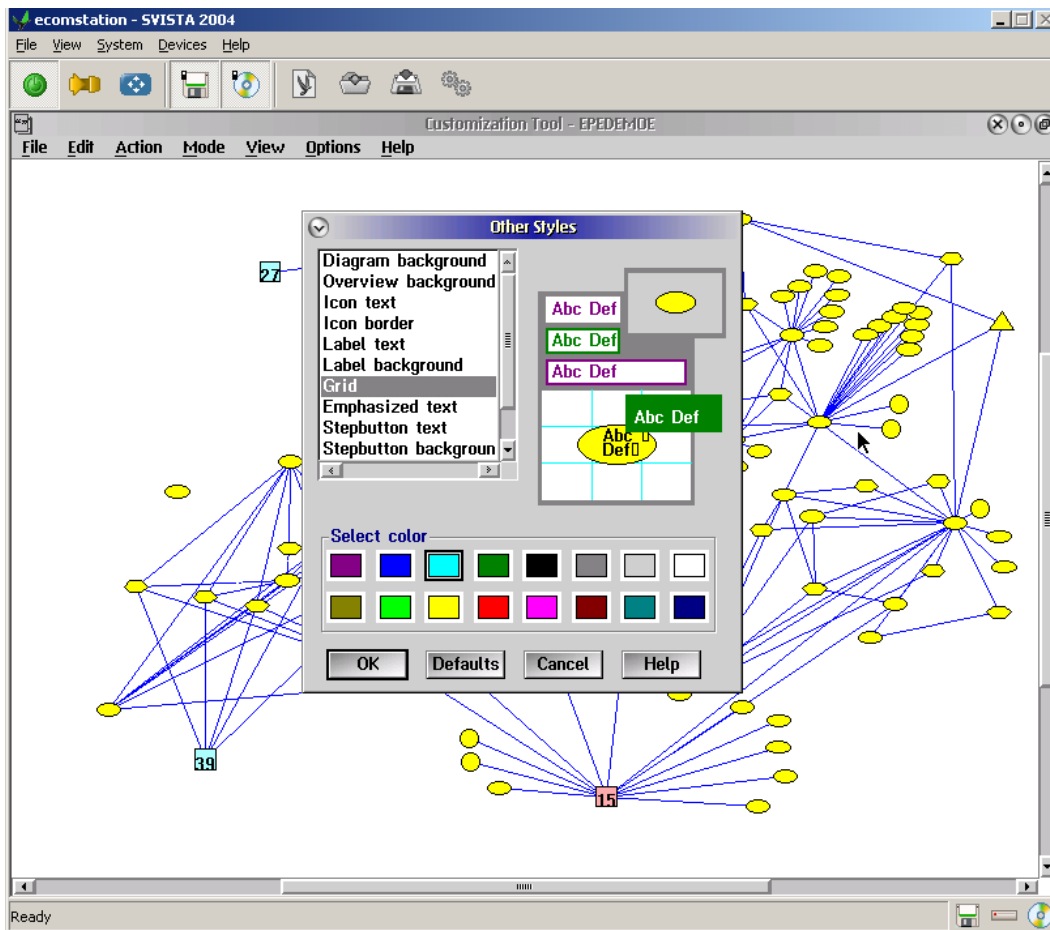


Figure 25. Defining styles of other diagram elements.

Use *Preferred* to save current styles as preferences or to replace current styles with the saved preferences.

CUSTOMIZATION TOOL USER'S GUIDE

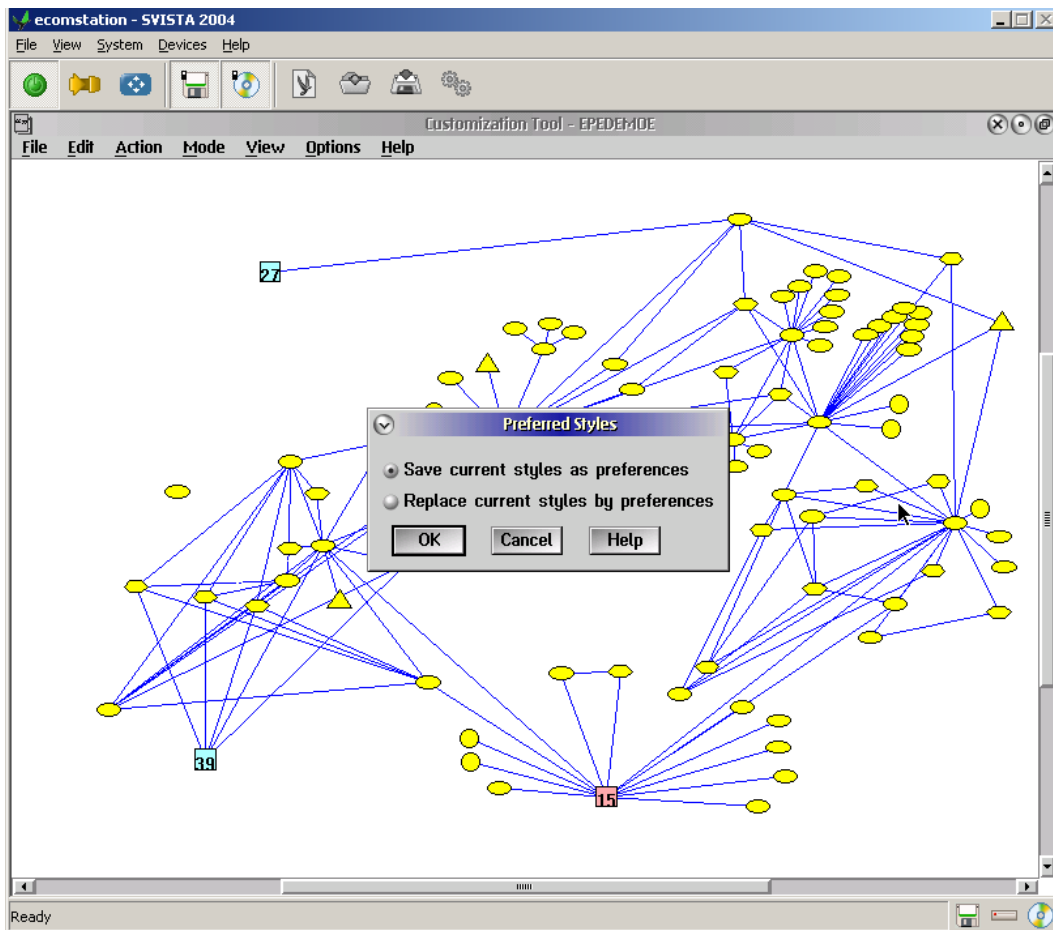


Figure 26. Saving and loading preferred styles.

10.37 Show labels

Use *Show labels* to display the name of each entity, cluster, table, or column when you select it.

This option is useful if the scale of your diagram is small and icon names are not displayed automatically.

CUSTOMIZATION TOOL USER'S GUIDE

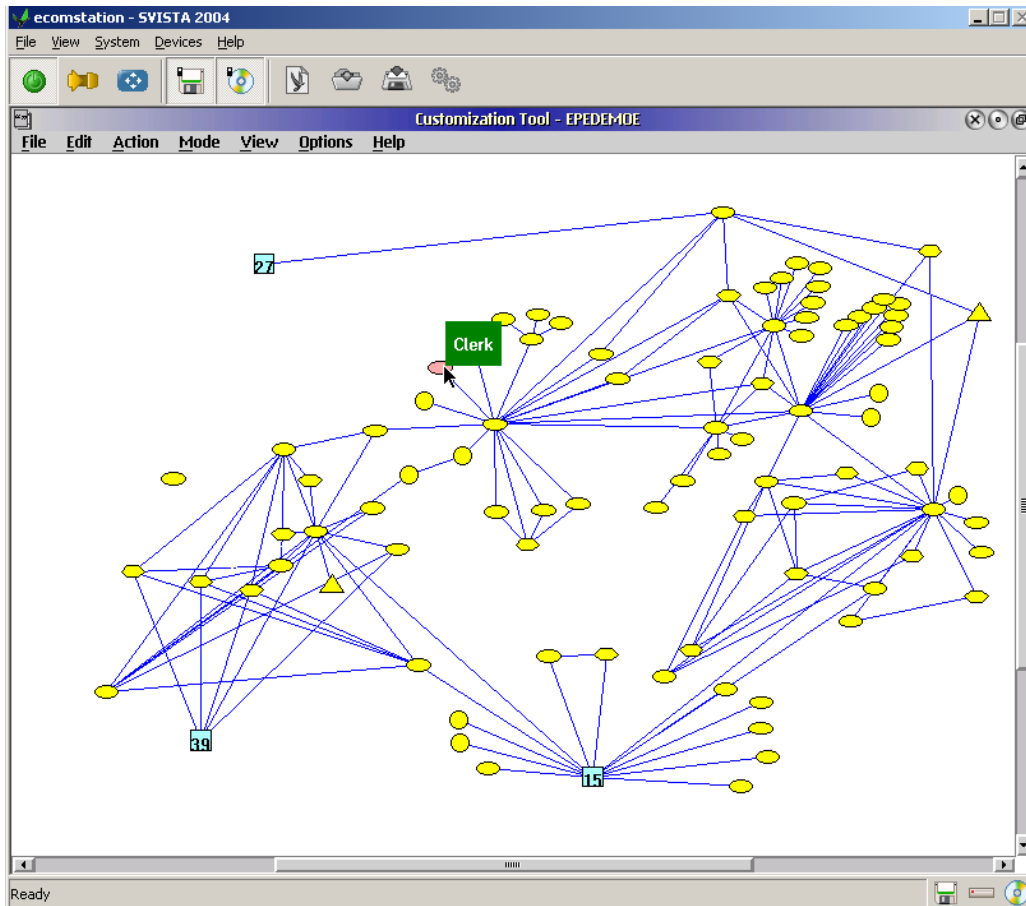


Figure 27. Showing entity label.

10.38 Show information area

Use *Show information area* to display the information area in the Define Entities window. The information area is the line at the bottom of the window that gives short help about the functions of the elements in the window.

Figure 52 on page 128 shows the *Define Entities* window with the information area showing.

10.39 Grid

Use *Grid* to change the grid setting.

You can select a visible grid, an invisible grid, or no grid. You can also select the size of the grid. A grid helps you align icons in the diagram.

CUSTOMIZATION TOOL USER'S GUIDE

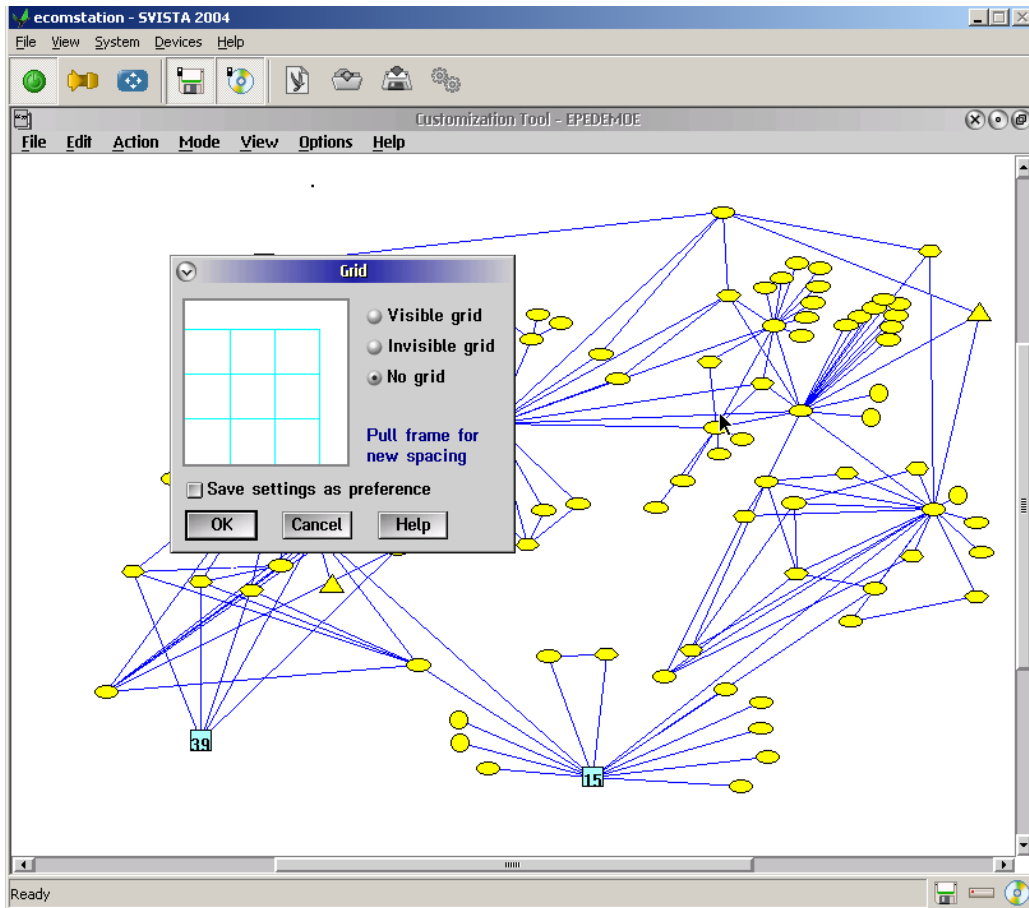


Figure 28. Defining a grid.

10.40 Name break

Use *Name break* to specify which characters are used to divide the text in the icons into two lines.

CUSTOMIZATION TOOL USER'S GUIDE

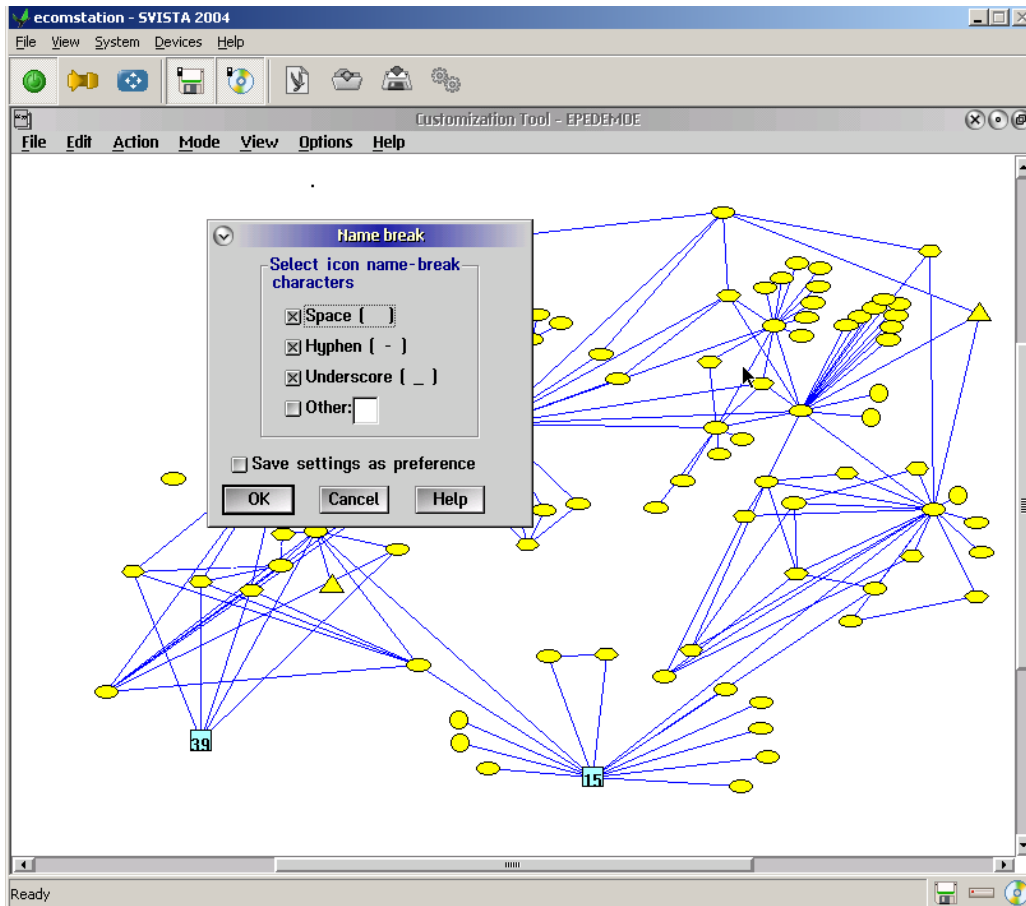


Figure 29. Defining name-break options.

Select *Space name break* to specify a space () as a name break.

Select *Hyphen name break* to specify a hyphen (-) as a name break.

Select *Underscore name break* to specify an underscore () as a name break.

Select *Other name break* to specify your own character as a name break. The default character, if you select this check box, is %.

You can change this character to your own preference by using the adjacent entry field:

10.41 Preferences

Use Preferences to define your preferred way of using the Customization Tool. You can set these options:

CUSTOMIZATION TOOL USER'S GUIDE

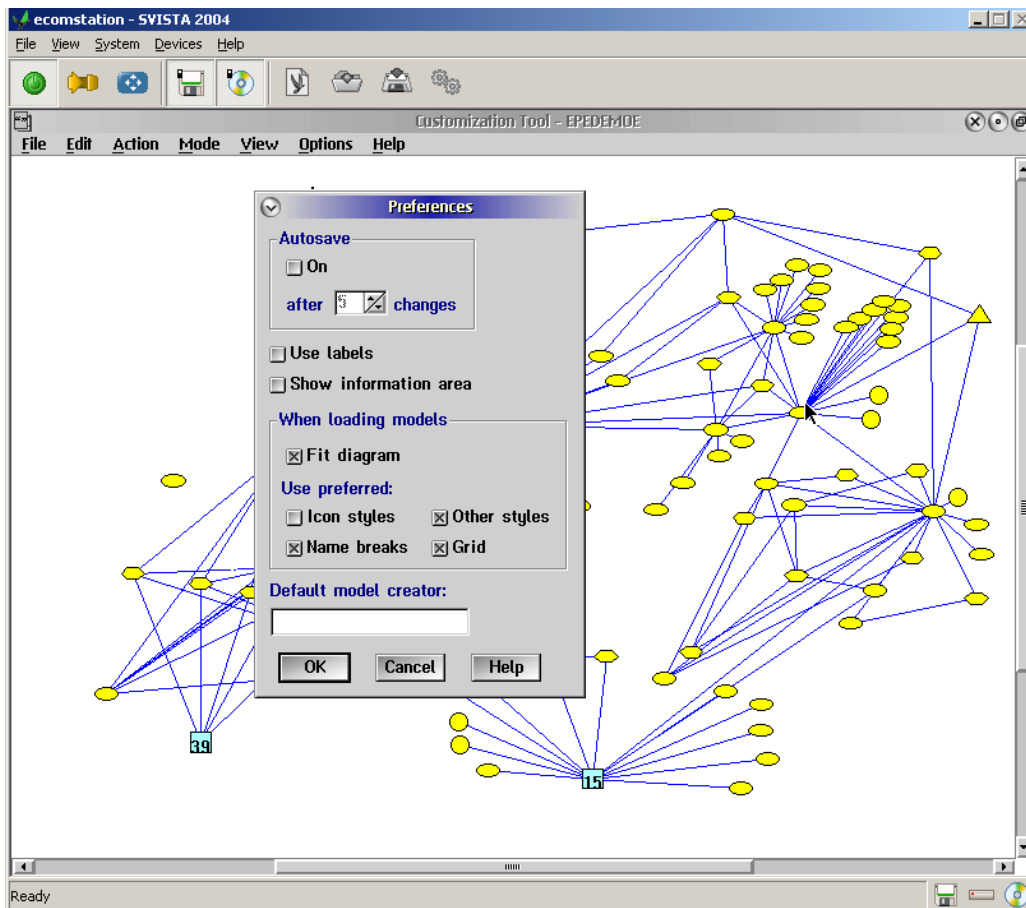


Figure 30. Defining preferences.

Autosave

Set autosave on and off and the number of changes between saves.

Use labels

State whether or not to use labels.

Show information area

State whether or not to show the information area on windows that have one.

Fit diagram

State whether or not to fit diagrams to the current window size when loading models or whether to use the scale factor that was in operation when the model was saved.

Icon styles

State whether to use preferred or the model icon styles when loading models.

Other styles

State whether to use preferred or the other model styles when loading models.

Name breaks

State whether to use preferred or the model name breaks when loading models.

CUSTOMIZATION TOOL USER'S GUIDE

Grid

State whether to use preferred or the model grid settings when loading models.

Default model creator

Specify the default model creator.

CUSTOMIZATION TOOL USER'S GUIDE

11 SPECIFYING ENTITIES

This chapter explains what features of the graphical user interface you need to define the entities in your conceptual model.

11.1 The language diagram

The language diagram is the basis for your work. When you finish the table preparations described in Chapter 4, "Working with the database" on page 22, and select *Language* from *Mode*, the language diagram appears. Initially, it contains a proposed entity for each table and column you have added, and proposed relationships between them. A proposed entity is represented by a circular icon, and a proposed relationship is represented by a dotted line in the diagram.

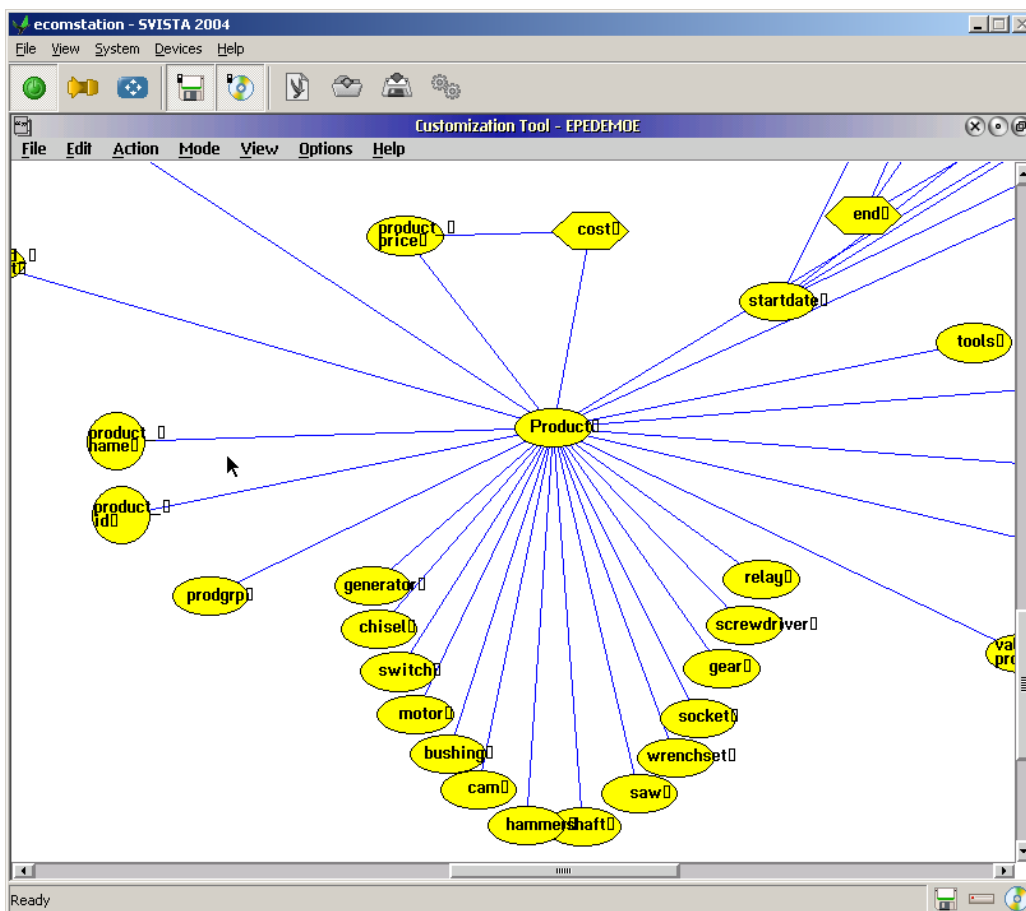
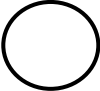


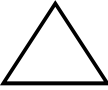




Figure 31. Language diagram.

The shapes of the icons and lines in the diagram change as you define the entities and relationships:

CUSTOMIZATION TOOL USER'S GUIDE

	An entity with no term
	An entity whose terms are nouns or proper names
	A verb entity
	An adjective entity
	A cluster of entities and clusters
	A relationship between two entities

11.2 Working with the Define entity window

Use the *Define entity* window¹ to define the name, class, term, syntax, and SQL of the entity.

¹ The *Define entity* window and the *Create entity* window are identical.

CUSTOMIZATION TOOL USER'S GUIDE

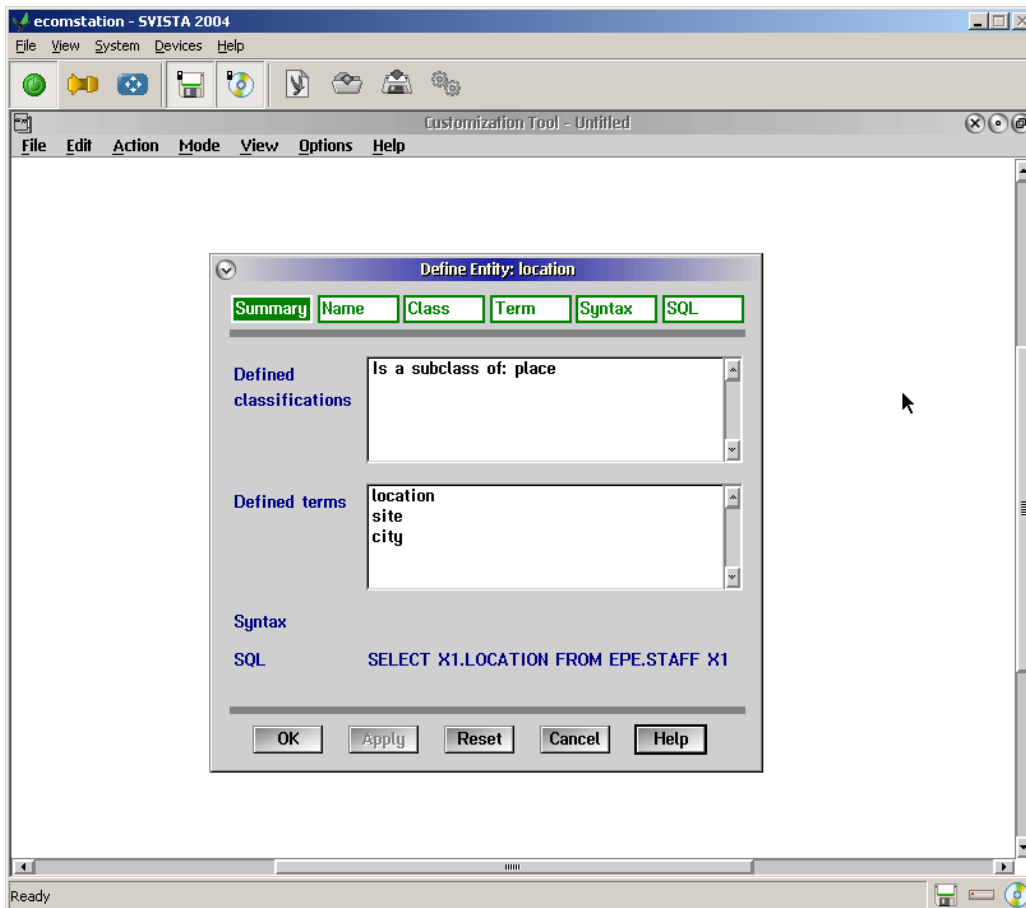


Figure 32. Define entity window.

The window title contains the name of the entity that you are working with. The window contains these buttons:

<i>Summary</i>	To display a summary of the existing definitions for the entity.
<i>Name</i>	To create or change the name of the entity.
<i>Class</i>	To classify the entity.
<i>Term</i>	To define the category of the entity and create or change terms.
<i>Syntax</i>	To specify the language syntax for the entity.
<i>SQL</i>	To create or change an SQL statement for the entity.
<i>OK</i>	To submit the input and remove the window.
<i>Apply</i>	To submit the input and keep the window.
<i>Reset</i>	To discard the input and keep the window.
<i>Cancel</i>	To remove the window without performing any action.
<i>Help</i>	To get help on any field or selectable item.

When you enter the Define entity window for an existing entity, Summary is selected. The window displays any existing definitions for the entity.

To change the information displayed in *Summary*, select *Name*, *Class*, *Term*, *Syntax*, or *SQL*.

CUSTOMIZATION TOOL USER'S GUIDE

11.3 Naming an entity

When you create a new entity, the Create entity window appears, with *Name* selected. The window contains an empty entry field, in which you enter a name for the entity.

To change the name of an existing entity, select *Name*. The existing name appears in the entry field.

Each table or column entity has a default name. The name is the same as the name of the table or column that the entity represents.²

Each entity name must be unique. It can contain up to 25 characters.

1. Type the entity name in the entry field.
2. Press *Apply* to submit the input and keep the window, or
3. Press *OK* to enter the input and remove the window.

² If two or more columns have the same name, the customization tool adds a number to the entity names to distinguish between the entities.

CUSTOMIZATION TOOL USER'S GUIDE

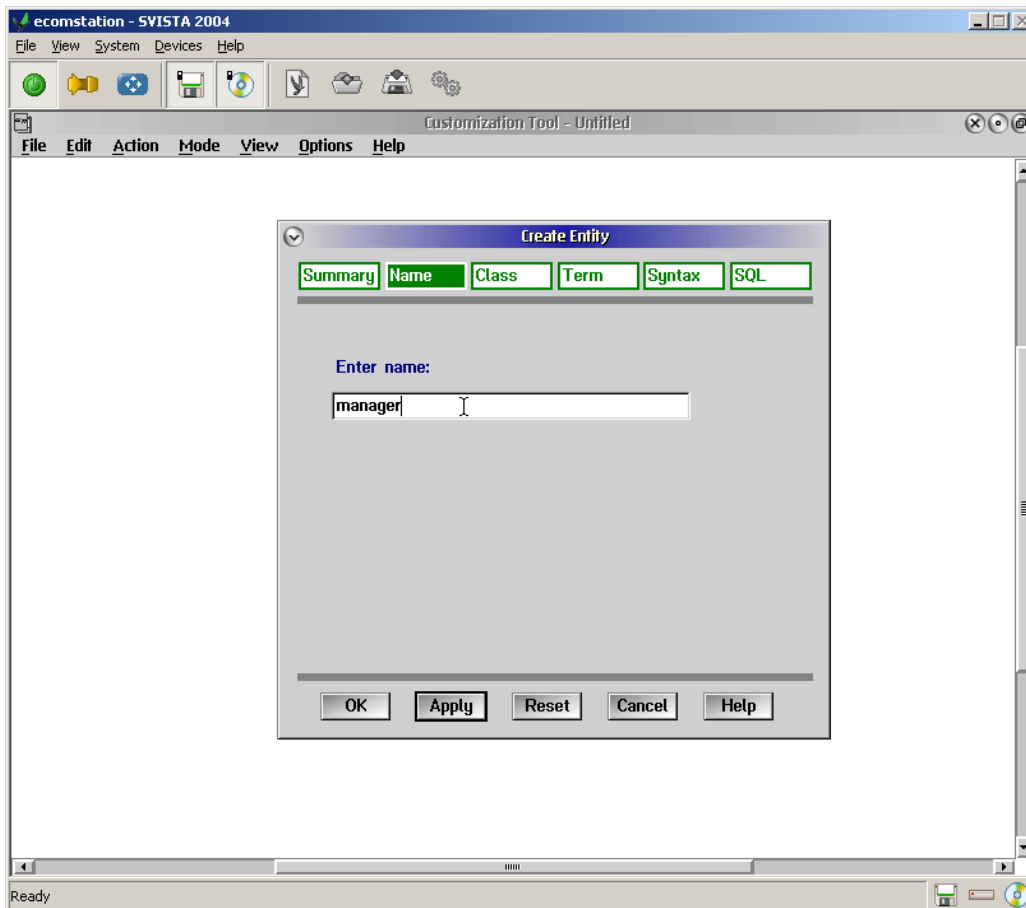


Figure 33. Specifying an entity name.

11.4 Classifying an entity

Classify each entity in your conceptual model as a subclass or instance of at least one other class.

For information about which class to choose, see "Classifying entities" on page 38.

11.5 Subclass and instance entities

1. Press the *Class* button to display the classification window:

CUSTOMIZATION TOOL USER'S GUIDE

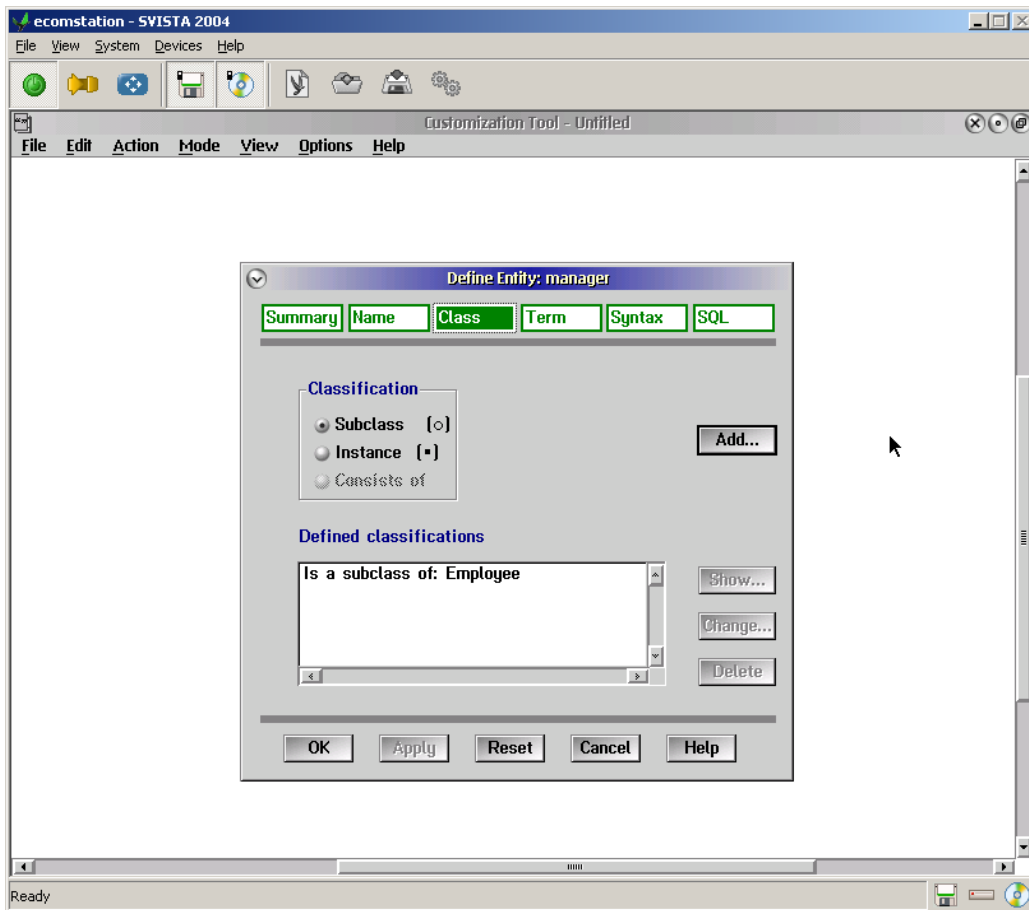


Figure 34. Classification window.

2. Select *Subclass* or *Instance*.
3. Press *Add* to display the hierarchy of classes:'

CUSTOMIZATION TOOL USER'S GUIDE

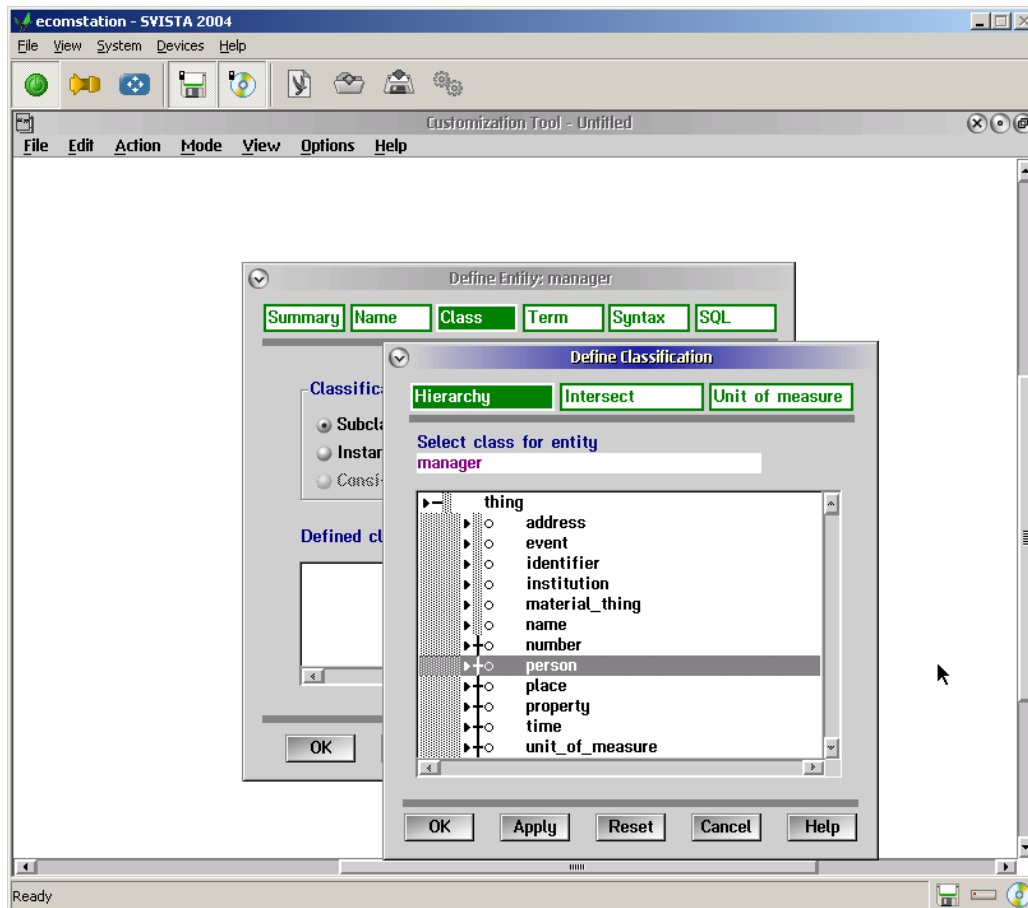


Figure 35. Class hierarchy.

Some classes in the hierarchy already have subclasses. To expand the hierarchy and view subclasses, double click on the **+**. To collapse the hierarchy and hide subclasses, double click on the **-**.

There is an icon in front of each class. The **O** icon represents a subclass; the **■** icon represents an instance.

4. Select the class under which you want to add your entity.

5. Press *Apply* to add the entity.

If the class you select is collapsed, a **+** appears in front of it to show that you can now expand that class. If you double click on it, you will see how your entity has been added to the hierarchy.

CUSTOMIZATION TOOL USER'S GUIDE

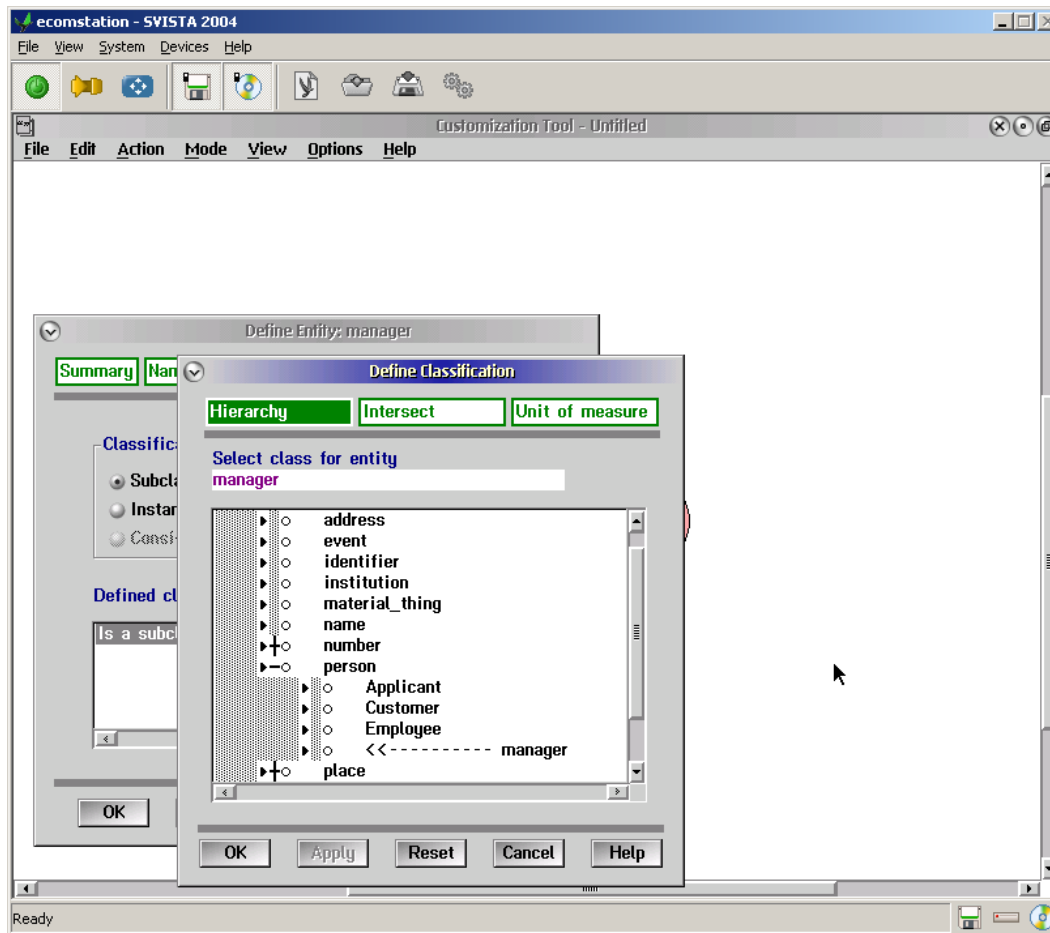


Figure 36. Adding a subclass to the hierarchy.

11.6 Composite entity

A *composite entity* is an entity that consists of a combination of two or more entities that have an SQL statement.

To classify a composite entity:

1. Press *Class*.
2. Add the entity as a subclass of one of the classes in the hierarchy.
3. Press *OK* in the Define classification window.
4. Select *Consists of* in the Define entity window, and press *Add*.
5. Select the constituent entities in the same order, as Ergo users will use them in their questions when they enter data values of the corresponding entities. Avoid more than 3 or 4 constituents for each composite entity. Otherwise questions involving composite entities may cause memory problems.

Each entity that you select becomes highlighted. The list in the lower part of the window shows the order in which you have selected the entities.

To remove a selected entity, click on it once in the list in the upper part of the window. The highlighting then disappears, and the entity is removed from the *Ordered selection* list.

CUSTOMIZATION TOOL USER'S GUIDE

6. Press OK.

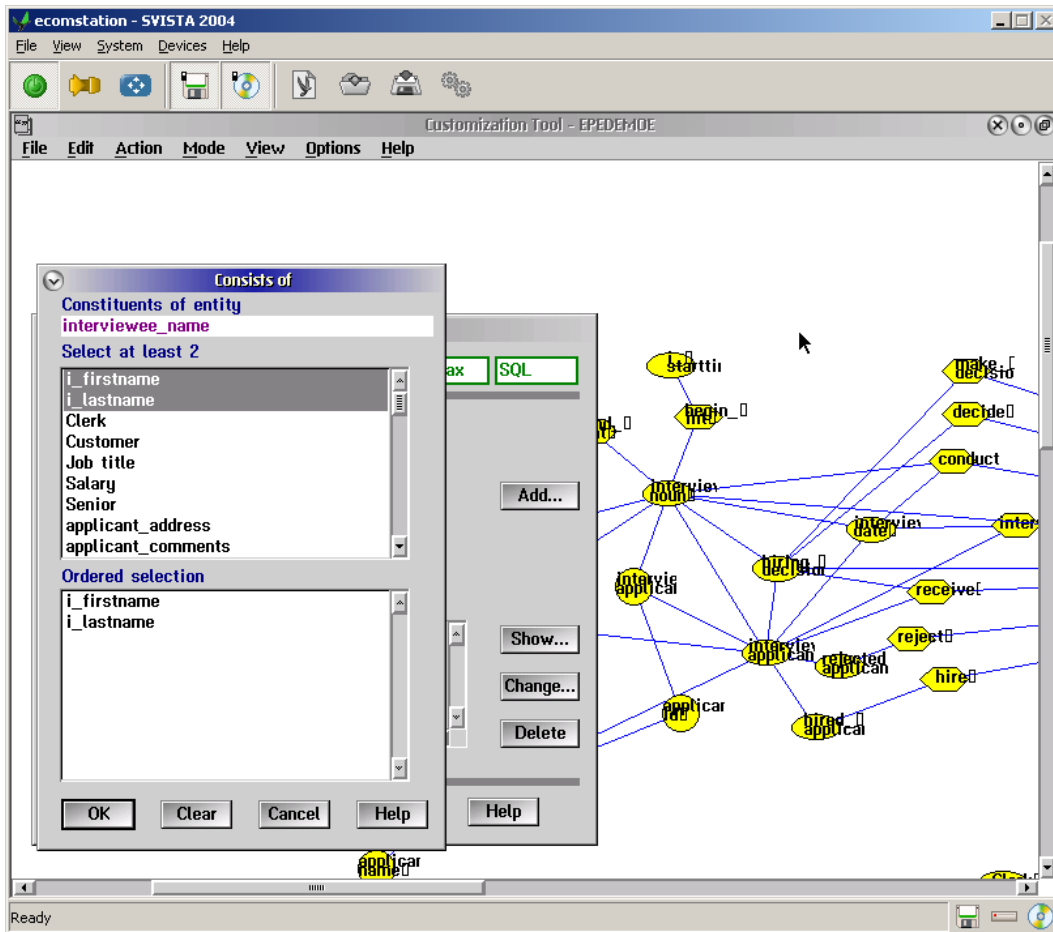


Figure 37. Selecting constituent entities.

An entity can have only one *Consists of* classification. For further information on composite entities, see "Composite entities" on page 61.

11.7 Intersecting entities

If two entities are subclasses of the same class, they can be either intersecting entities or disjoint entities.

Intersecting entities represent concepts that can be used in combination. In EPEDEMOE, the sample conceptual model, clerk and senior are intersecting entities because a person can be a clerk and also senior.

Disjoint entities represent concepts that are mutually exclusive. In EPEDEMOE, clerk and manager are disjoint entities because a person cannot be both a clerk and a manager.

By default:

- All subclasses of a predefined class in the hierarchy are disjoint entities.
- All subclasses of an entity that you define are intersecting entities.

CUSTOMIZATION TOOL USER'S GUIDE

To specify disjoint entities:

1. Double click on the icon for one of the entities.
2. Press Class.
3. Select the class in the Defined classifications list.
4. Press Change to display the Define classification window.
5. Press Intersect to display a list of intersecting entities.
6. Select the entities that are disjoint with your entity.
7. Press OK.

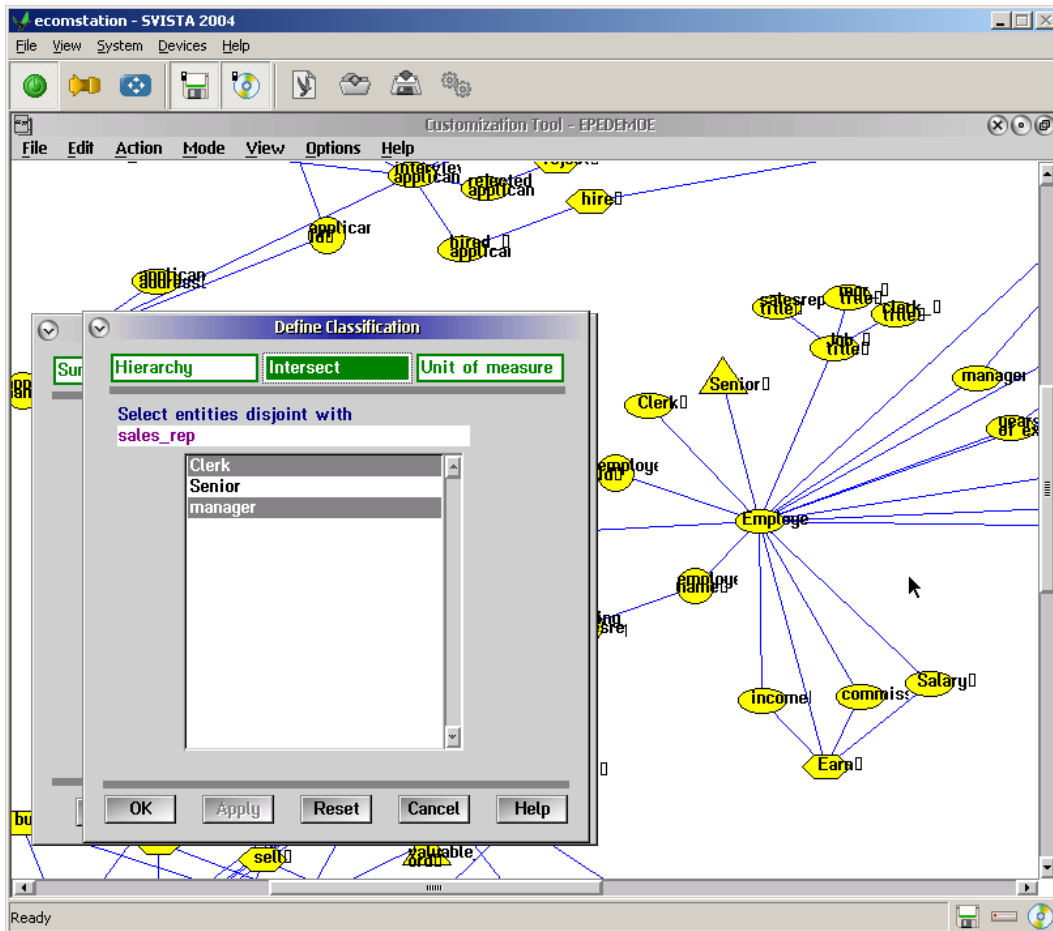


Figure 38. Selecting disjoint entities.

If you specify that entity A is disjoint with entity B, you need not specify that entity B is disjoint with entity A.

11.8 Units of measure

When you classify an entity as a subclass of *quantified_property* (or a subclass of a subclass of *quantified_property*), you can specify its unit of measure. The Customization Tool contains units of measure for age, area, currency, duration, length, volume, and weight. When you specify a unit of measure, users can use that unit of measure in their questions, for example:

How many dollars does a chisel cost?

CUSTOMIZATION TOOL USER'S GUIDE

11.8.1 Selecting an existing unit of measure

To select an existing unit of measure:

1. Classify the entity under *quantified_property* (for example, as a subclass of price).
2. Press *Apply*.
3. Press *Unit of measure* in the *Define classification* window.
4. Select the unit of measure that you want from the list (for example, dollar).
5. Press *OK*.

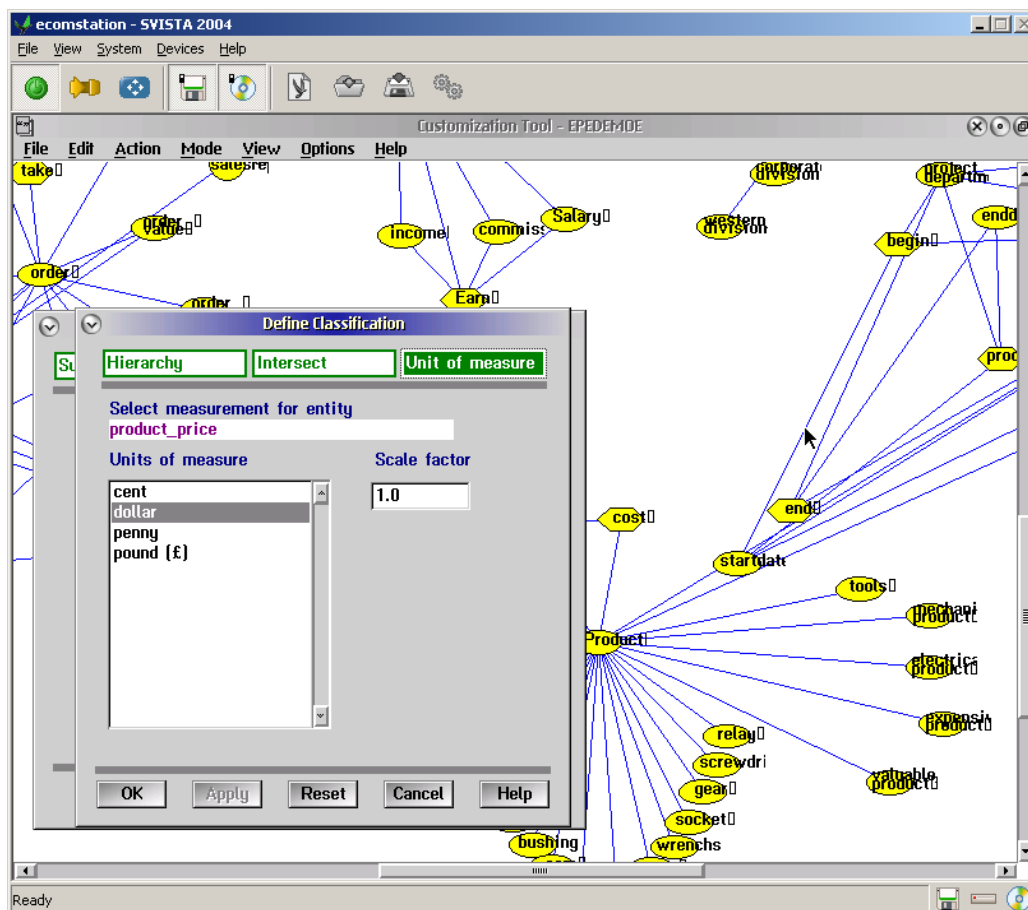


Figure 39. Specifying units of measure.

You can also specify a scale factor. For example, if your database contains weights measured in pounds, but users instead want to talk about ounces in their questions, select ounce as the unit of measure and specify 16 as the scale factor because 16 ounces = 1 pound.

Note that the scale factor is used only to convert the unit of measure in a question so that it corresponds to the unit of measure used for data values in the column. The answer to a question is not converted.

11.9 Changing or deleting an existing classification

Sometimes you want to change or delete an existing classification, or you want to see where an entity fits in the hierarchy. To change or delete a classification, or to show the entity in the hierarchy:

1. Select *Class* in the *Define entity* window.

CUSTOMIZATION TOOL USER'S GUIDE

2. Select a classification in the Defined classification list box.
3. Use one of these buttons:
 - Change* To change the classification. This button displays the hierarchy. Select the class under which you want to add the entity.
 - Delete* To delete the classification.
 - Show* To show how the entity fits into the hierarchy of classes.

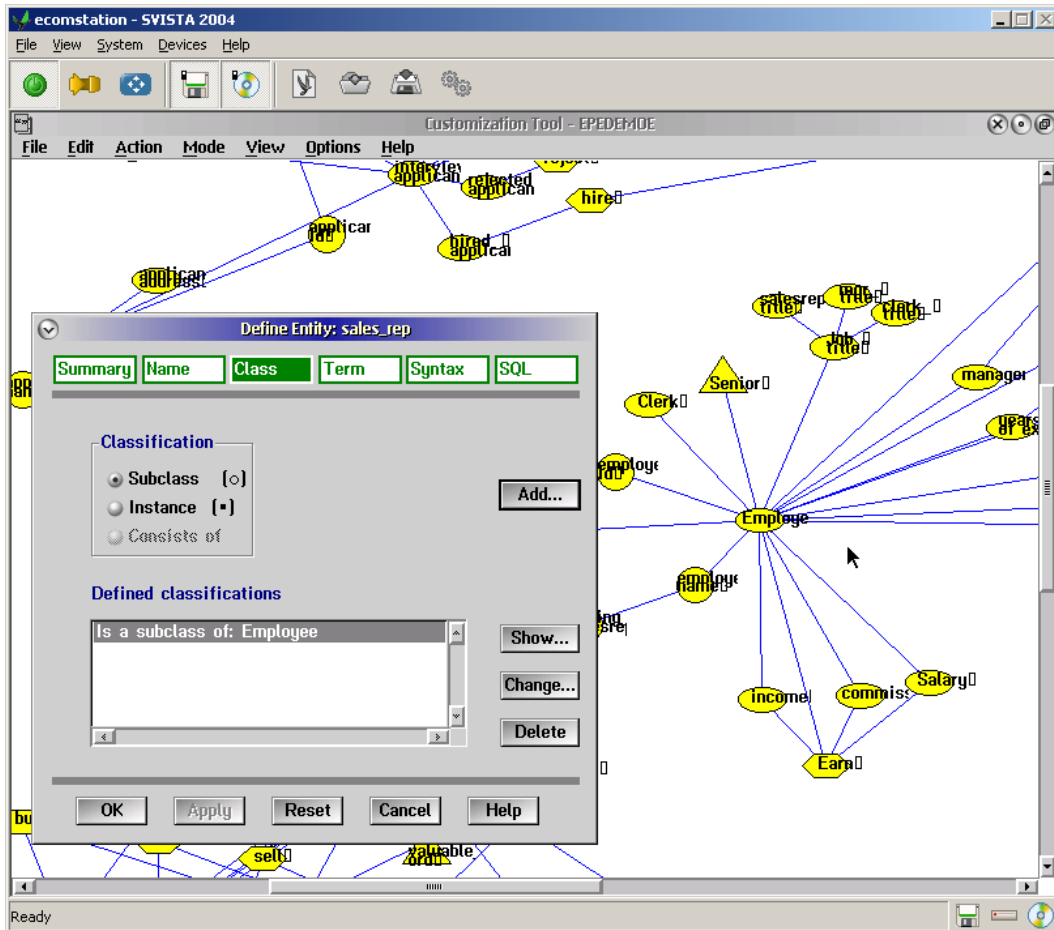


Figure 40. Changing a classification.

Note that when you change or delete the classification of an entity for which you have already defined relationships, these relationships may become invalid. "Correcting invalid relationships" on page 139, explains how to correct an invalid relationship.

11.10 Creating terms

Create one or more terms for each entity that users want to refer to.

The terms you create can be nouns, verbs, adjectives, or proper names. Each term that refers to the same entity must have the same category; if you give an entity a term that is a noun, the additional terms you create for that entity are automatically nouns.

The first term that you create is considered the primary term. It will appear in the interpretations that users see in response to their questions. Create the most descriptive term first, and then create additional terms.

CUSTOMIZATION TOOL USER'S GUIDE

You can use the same term for more than one entity. If possible, avoid using the same primary term for two entities because this could result in ambiguous interpretations.

A term can consist of one or more words; it can be up to 80 characters long.

Note: Do not confuse terms with entity names. The name is used to distinguish the entity icon in the diagram. Terms are used in questions.

1. To create terms:
2. Press *Term* in the *Define entity* window.
3. Enter your term: for example, *employee*.
4. Select a category: Noun, Verb, Adjective, or Proper name.
5. Press *Add*.

If your term is a noun, adjective, or verb, a window appears where you specify the grammar of the term.

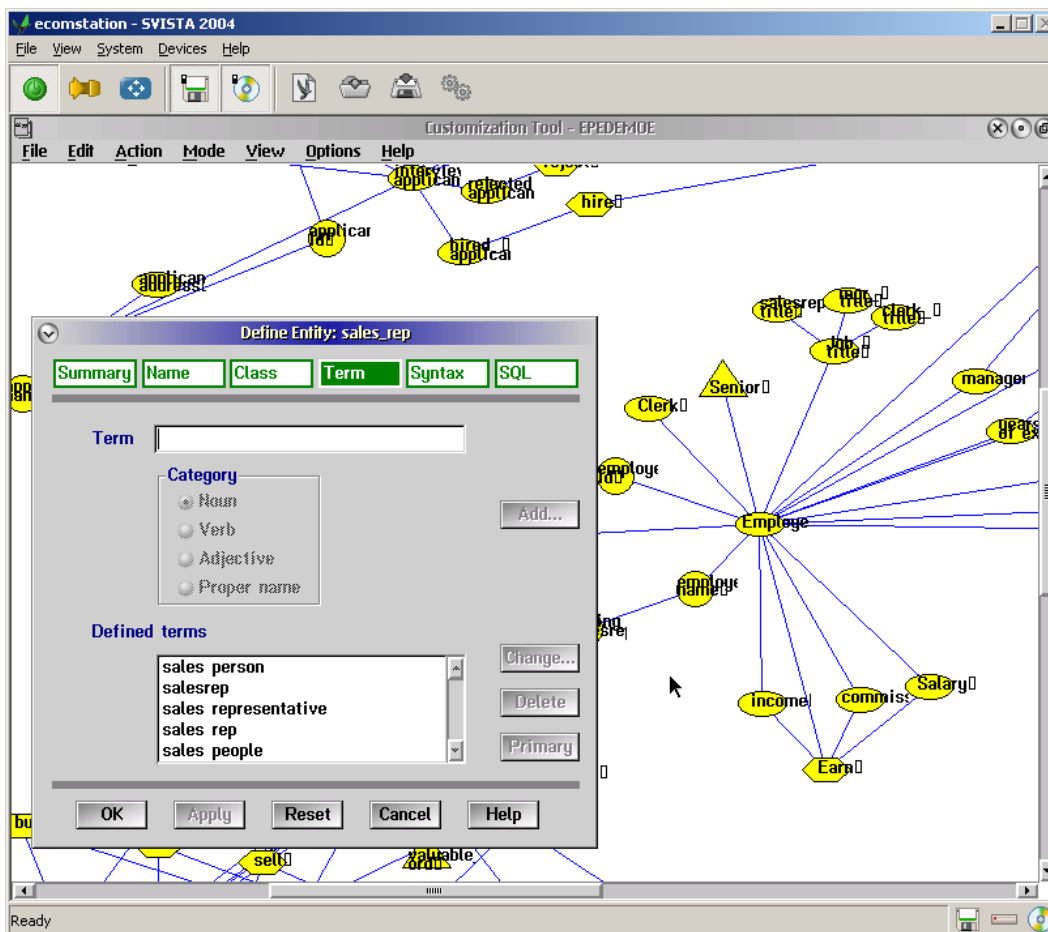


Figure 41. Creating a term.

To change the primary term:

1. Select the term in the list that you want to make the primary term.
2. Press *Primary*.

CUSTOMIZATION TOOL USER'S GUIDE

11.10.1 Noun grammar

When you add a noun, the Noun grammar window appears.

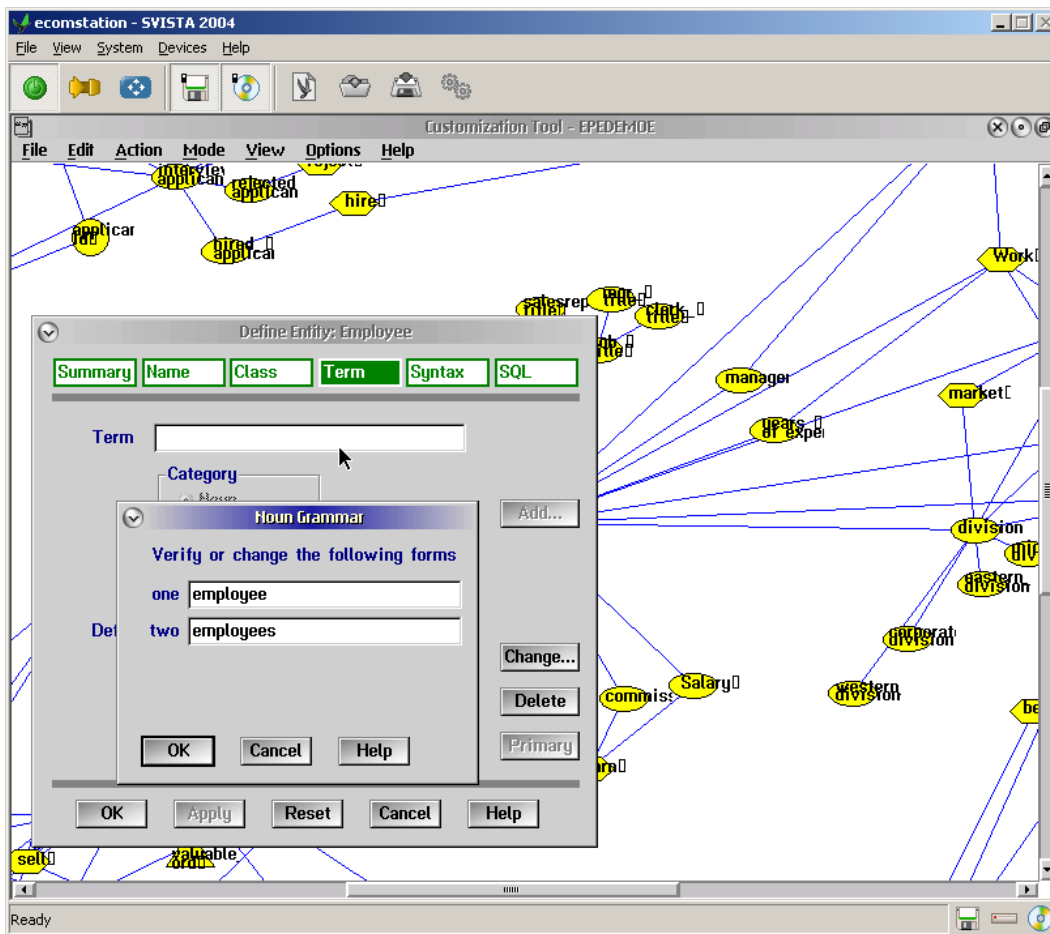


Figure 42. Specifying singular and plural forms of a noun.

1. Specify the singular and plural forms of your noun. If your noun is used only in one of the two forms, blank out the proposed form that is incorrect or redundant. For example, the noun traffic is used only in its singular form, and you should blank out the proposed plural form traffics.
2. Press OK.

If you specify a singular form, the *Select Pronouns* window appears.

CUSTOMIZATION TOOL USER'S GUIDE

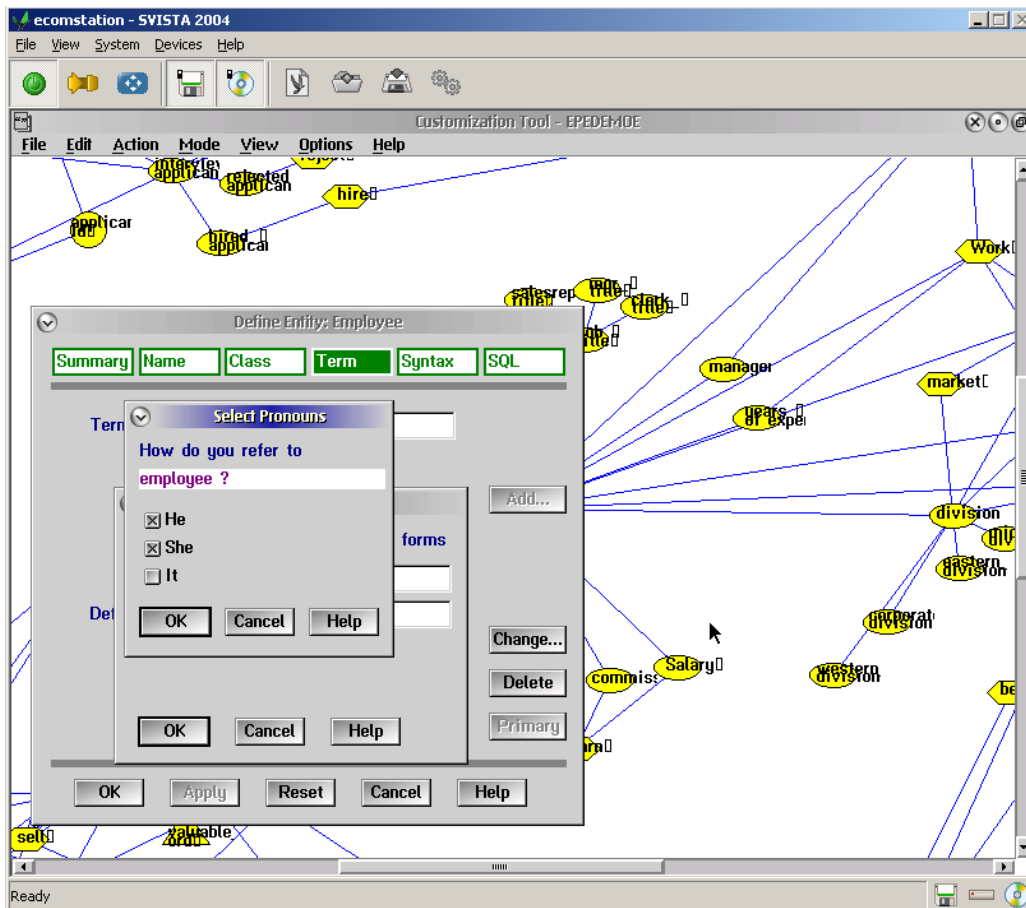


Figure 43. Specifying pronouns referring to a noun.

3. Select all of the pronouns that can be used when referring to the noun.
4. Press OK.

Compound terms: The Compound term window appears when you enter a noun that contains two or three words.

CUSTOMIZATION TOOL USER'S GUIDE

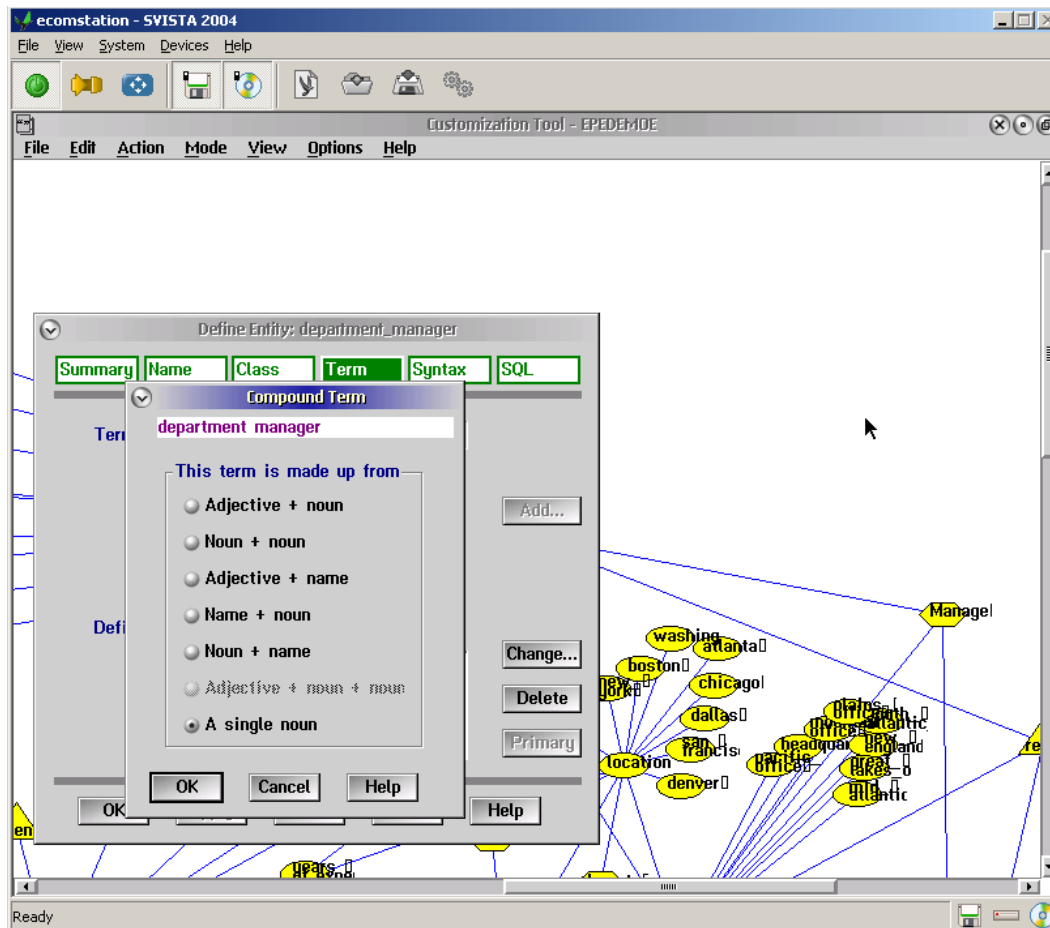


Figure 44. Specifying a compound term.

1. Select the categories of your compound term.

If you select a single noun, all the words that make up the term must appear together in user questions. If you specify something else, users can use the words separately in a question. For example, if you enter the term electrical product, and specify it as an adjective + noun, users can ask:

*Which are the electrical products? or
Which products are electrical?*

2. Press *OK*.
3. For each part of the term that is a noun, specify the singular and plural forms, and the referring pronouns in the dialog boxes that appear.

11.10.2 Verb grammar

When you define a term as a verb, the *Verb grammar* window appears.

1. Change or verify the suggested forms of your verb. Specify all five forms.
2. Press *OK*.

CUSTOMIZATION TOOL USER'S GUIDE

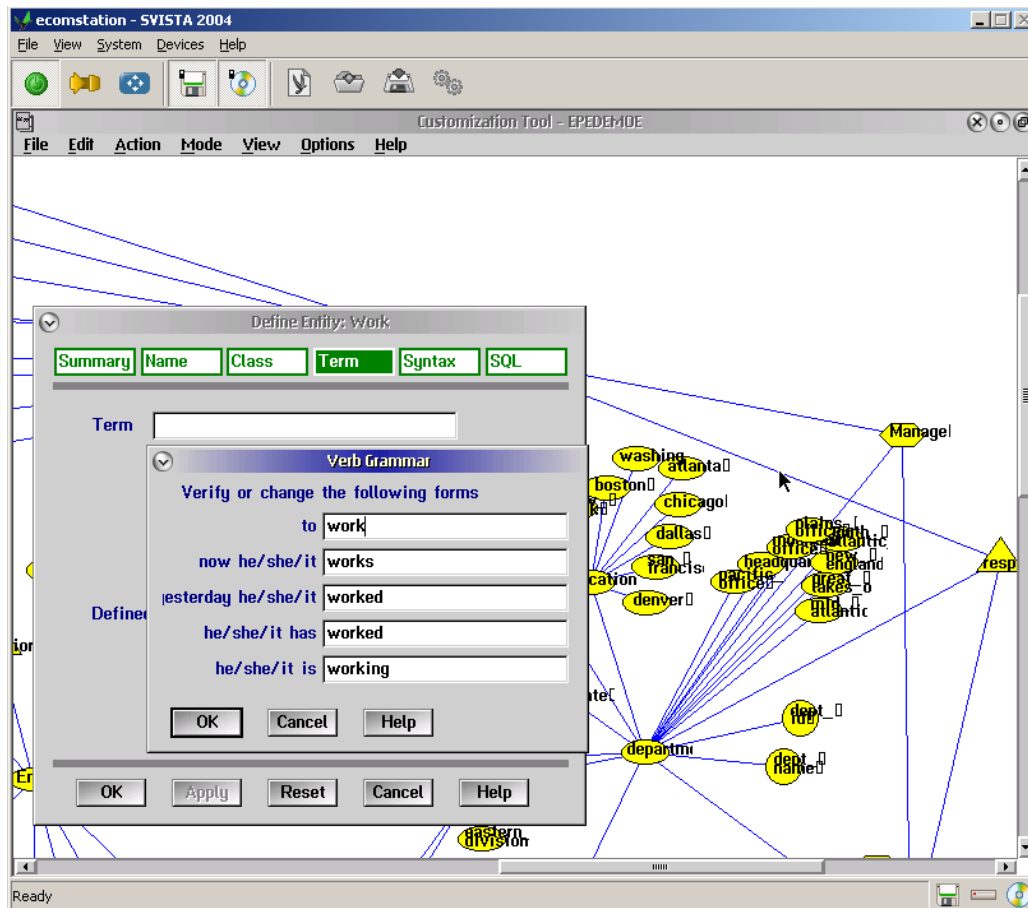


Figure 45. Specifying verb forms.

11.11 Specifying syntax

In the Customization Tool, syntax refers to how terms are used in a natural language question. You must specify syntax for each verb entity. Syntax is optional for noun and adjective entities.

Note that you specify the syntax for an entity, not for a term. If you create two or more terms for an entity, they have the same syntax.

The syntax of each part of speech is defined by the prepositions that are associated with each entity. You can select:

- Four prepositions for each noun
- Four prepositions for each adjective
- Four prepositions for each verb without a direct object
- Two prepositions for each verb with a direct object
- No prepositions for a verb with two direct objects

A question can contain only two prepositions (or one preposition and a direct object) for each entity. After you select the prepositions for an entity, Ergo asks you, which prepositions can appear in the same question (unless you select only one preposition, or the entity has a direct object).

CUSTOMIZATION TOOL USER'S GUIDE

11.12 Syntax for nouns and adjectives

You can optionally specify syntax for adjectives and nouns.

1. Press *Syntax* in the *Define entity* window.
The Preposition complement window appears.
2. Select the prepositions that you want to use with your entity.
3. Press OK.
4. If you select more than one preposition, the Syntax window appears with proposed phrases using your term. Select the phrase that suits your term.
5. Press OK

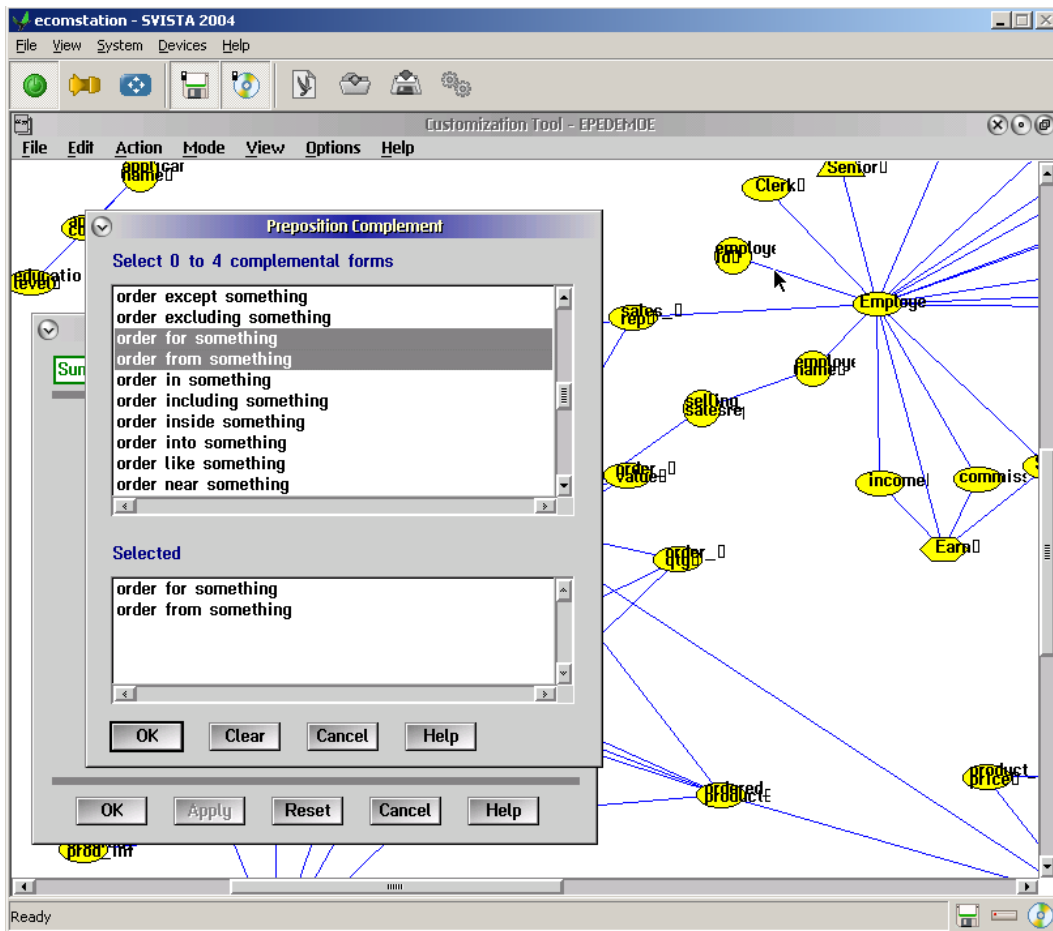


Figure 46. Specifying prepositions for nouns.

CUSTOMIZATION TOOL USER'S GUIDE

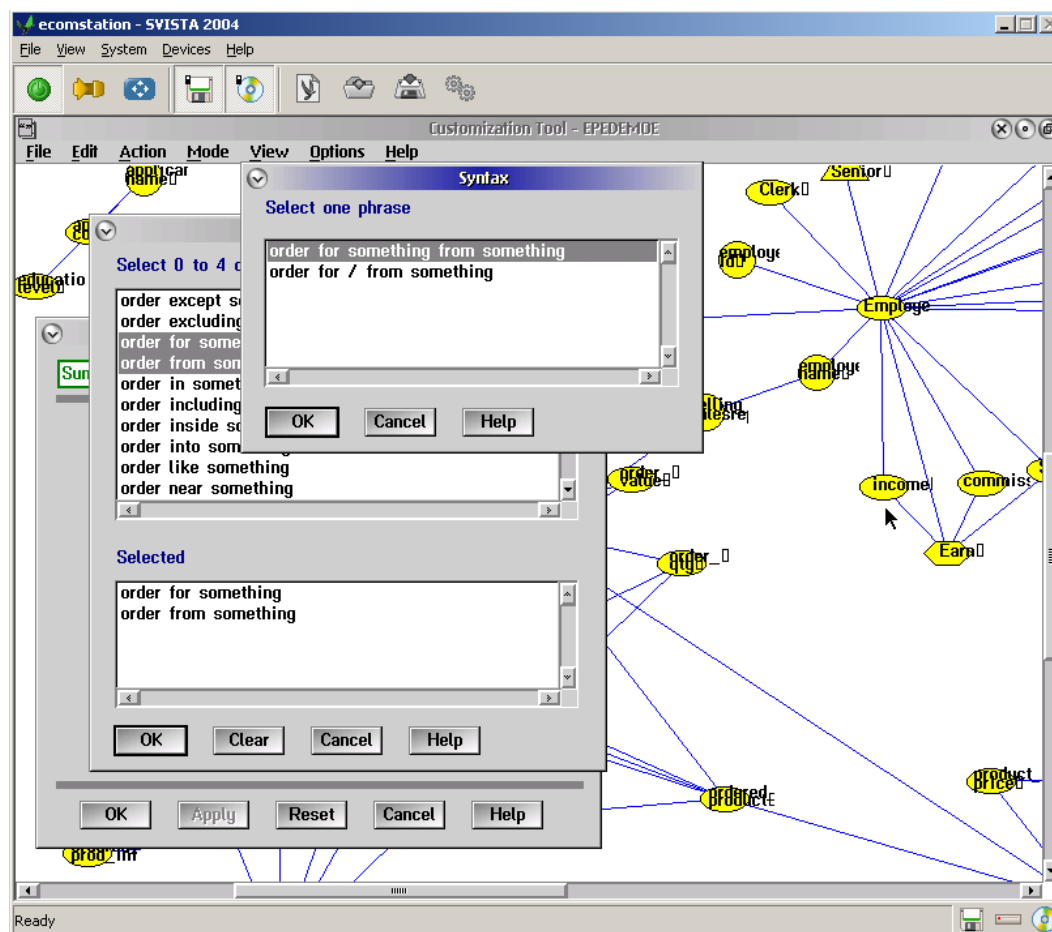


Figure 47. Specifying preposition syntax for nouns.

You can select any of these prepositions:

about	beyond	onto
above	by	opposite
according to	by way of	out
across	concerning	outside
after	down	over
against	during	per
ahead of	except	since
among	excluding	starting
apart from	for	through
around	from	to
as	in	toward
aside from	including	under
at	inside	until
away from	into	up to
before	like	versus

CUSTOMIZATION TOOL USER'S GUIDE

behind	near	via
below	next to	with
beneath	of	within
beside	off	without
between	on	

Table 4 Prepositions

11.13 Verb syntax

When you specify the syntax of a verb, you select a phrase that corresponds to how your verb is used in a sentence. For some verb phrases, you can also select prepositions that you want to use with the verb.

1. Press *Syntax* in the *Define entity* window.

The *Verb complement* window appears. It contains four alternative phrases with your verb:

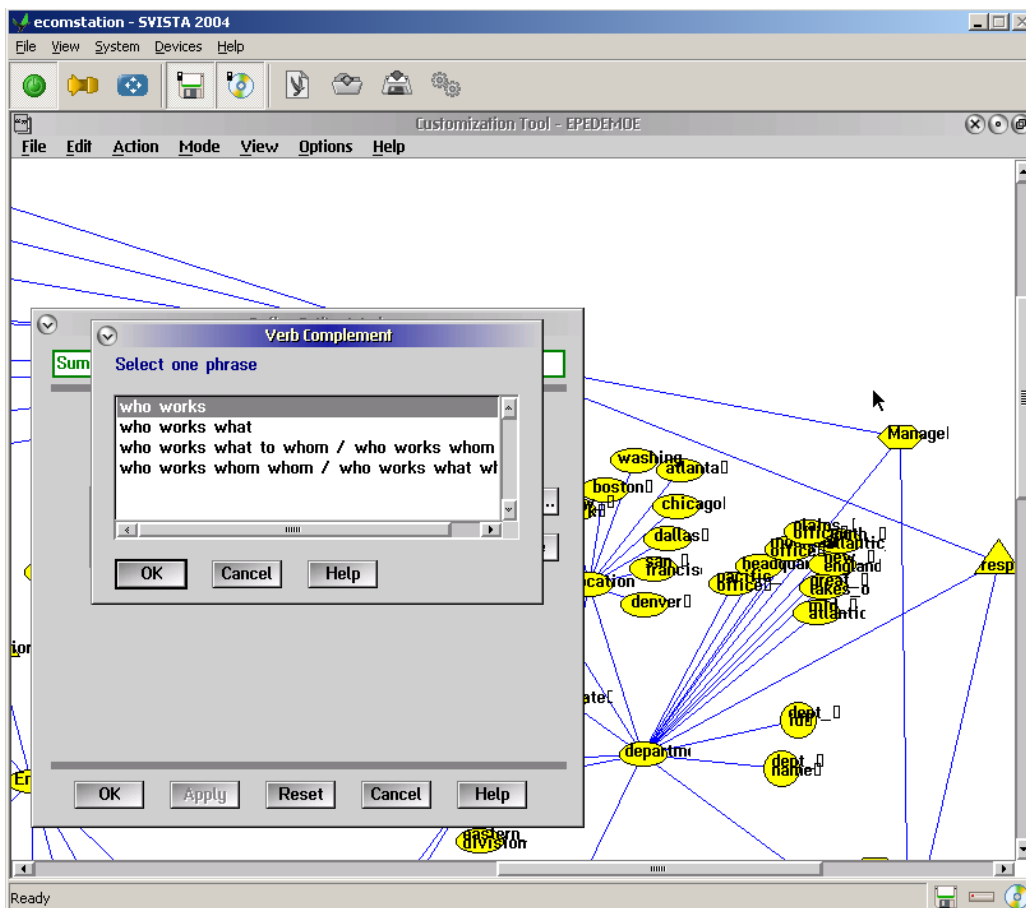


Figure 48. Specifying subject/object form for verbs.

2. Select the phrase that corresponds to how your verb is used in a question:

CUSTOMIZATION TOOL USER'S GUIDE

Without a direct object: For example, if you have created the verb *work*, and you want to use it in this context: *an employee works*, select the first phrase in the list:

who works

For this type of verb, you can select four prepositions.

With one direct object: For example, if you have created the verb *buy*, and you want to use it in this context: *a customer buys products*, select the second phrase in the list:

who buys what

Use this type of phrase also for verbs that you use in the passive voice (with no subject), such as *departments are located*.

For a verb with a direct object, you can select two prepositions.

With one direct object and one indirect object: or example, if you have created the verb *sell*, and you want to use it in this context: *a salesrep sells products to customers* or *a salesrep sells customers products*, select the third phrase from the list:

who sells what to whom/who sells whom what

For this type of verb, you can select two prepositions.

*With two direct objects:*³ For example, if you have created the verb *name*, and you want to use it in this context: *the ship-owner names the ship Isabella*, select the fourth phrase in the list:

who names whom whom/who names what what

This type of verb cannot have prepositions.

³ Verbs with two direct objects are rare. It is much more common to have a direct object and an indirect object.

CUSTOMIZATION TOOL USER'S GUIDE

3. Press OK.

If you select a phrase without a direct object, or with one direct object, the Preposition complement window appears when you press OK:

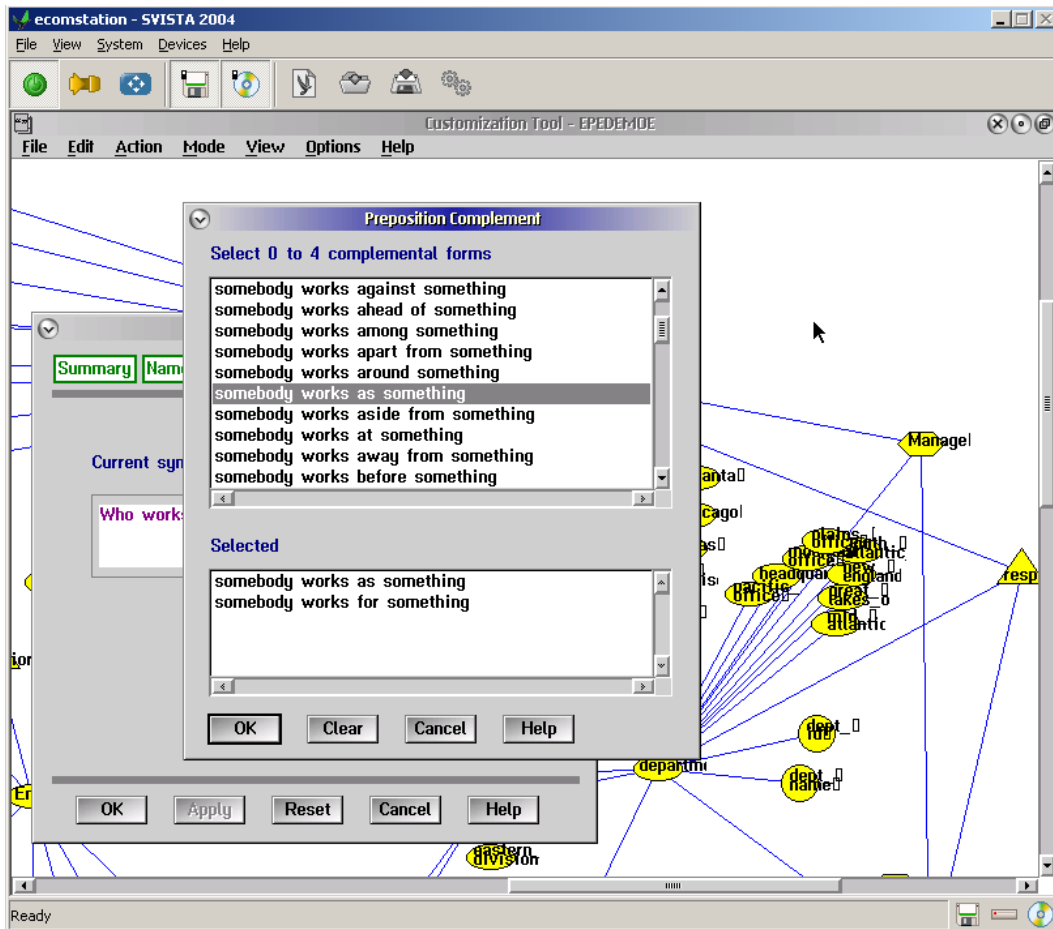


Figure 49. Specifying prepositions for verbs.

4. Select the prepositions that you will use with the verb. They are the same as the prepositions for nouns and adjectives, listed in Table 4 on page 123.

For example, if you are creating the verb work to use in expressions such as: *an employee works on a project for a manager*, select these two prepositions:

somebody works as something
somebody works for something

5. Press OK.

6. If you select more than one preposition, the Syntax window appears with proposed phrases using your term. Select the phrase that suits your term.

CUSTOMIZATION TOOL USER'S GUIDE

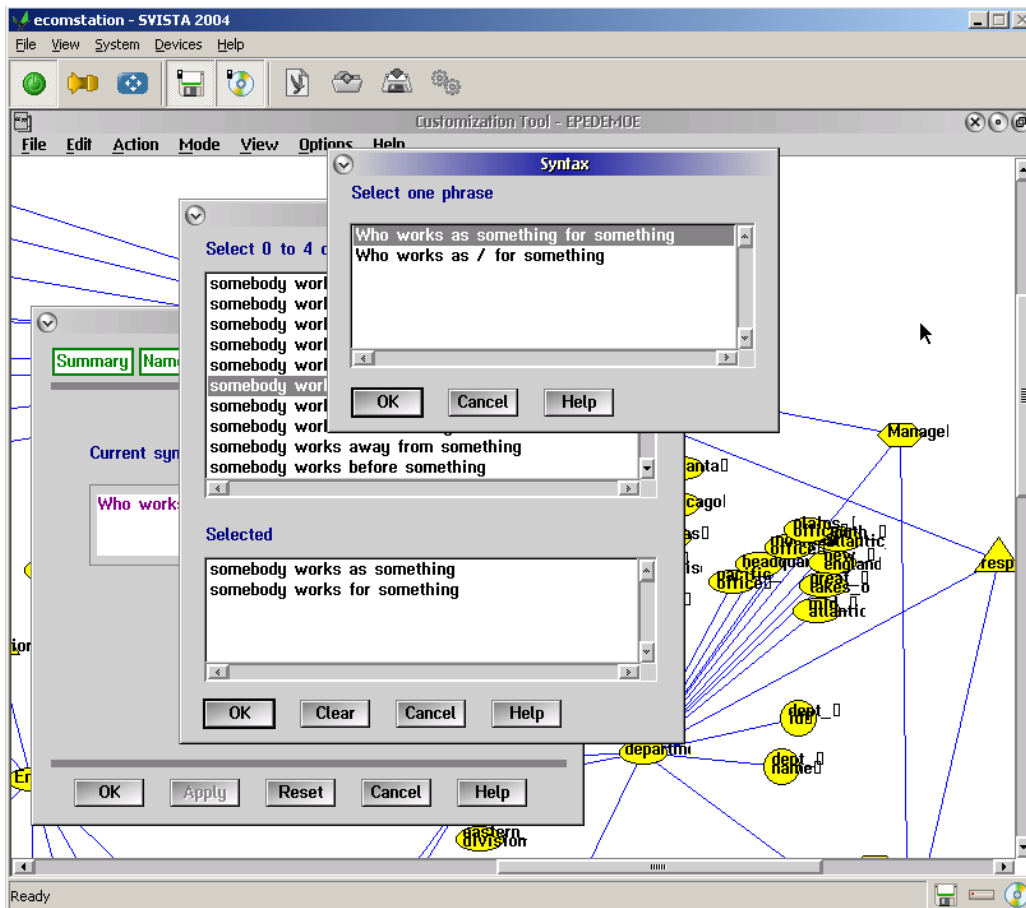


Figure 50. Specifying prepositions syntax for verbs.

7. Press OK.

11.14 Writing SQL statements

Use SQL to:

- Define the SELECT statement for an instance, adjective, or noun entity that you create yourself.
- Change the proposed SELECT statement for a column entity.

The Customization Tool proposes a SELECT statement for" each column entity. These are simple statements, such as:

```
SELECT X1.COMM  
FROM EPE.STAFF X1
```

If the column contains NULL values, you can change the SELECT statement so that the NULL values are disregarded in the answer to a question:

```
SELECT X1.COMM  
FROM EPE.STAFF X1  
WHERE X1.COMM IS NOT NULL
```

To specify an SQL SELECT statement:

1. Press SQL in the Define entity window.

CUSTOMIZATION TOOL USER'S GUIDE

2. Type the SELECT statement in the entry field, or use the Presentation Manager clipboard to copy text into the window from an application program.
3. Press *Apply* to enter the input and keep the window, or press *OK* to enter the input and remove the window.

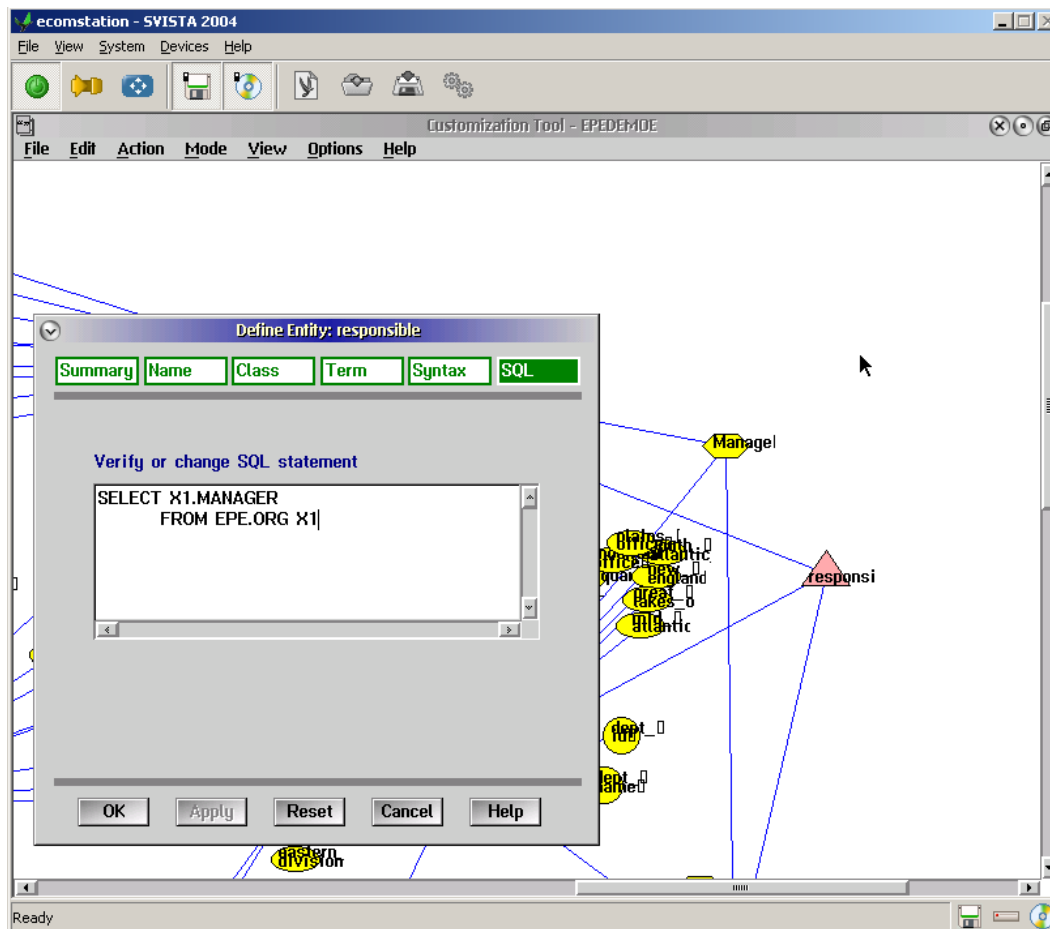


Figure 51. Specifying SQL for an adjective.

Note that if you modify the SQL statement for a column entity, go to database mode, then return to language mode, the Customization Tool generates a new entity with the original SQL statement. The column entity with a modified SQL statement remains unaffected in the conceptual model.

Only a subset of the SQL syntax is available for entities. See Appendix A, "SQL syntax in the Customization Tool" for a complete description of the SQL syntax that is available.

This section describes the form of SQL statements that may be used to specify the database representation of entities.

11.15 Handling multiple entities

If you select a group or cluster of entities when defining entities, you will be presented with the Define Entities window. This lets you define the characteristics of several entities from just one window, using values suggested by the Customization Tool or modified by yourself.

CUSTOMIZATION TOOL USER'S GUIDE

Figure 52. Specifying multiple **entities** on page 128 shows the multiple entities window. The spreadsheet in the center of the window contains suggested values for the various properties of the entities.

The name and SQL statement are defined automatically when entering language mode from database mode. The class, term, and category contain suggested values and are emphasized in a different color from the values that have already been defined.

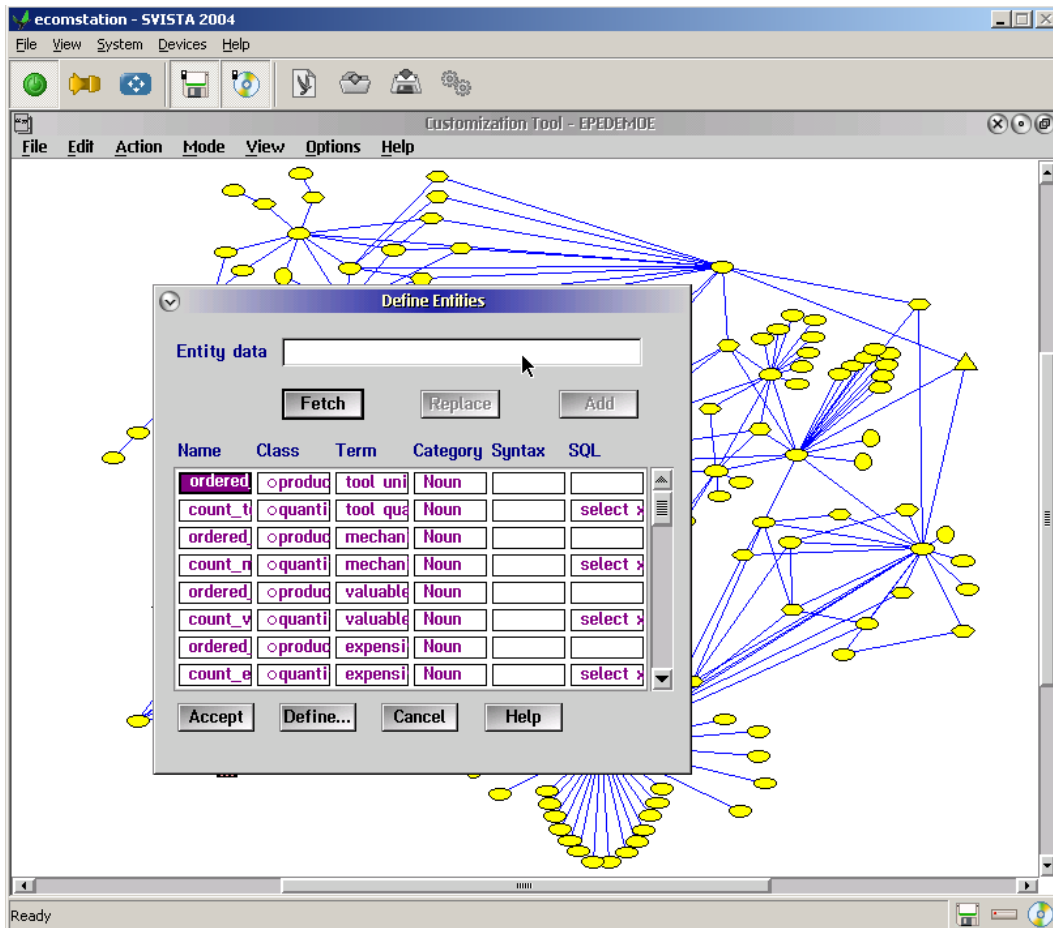


Figure 52. Specifying multiple entities.

The spreadsheet of cells contains one line for each entity in the selected cluster or group of icons. The columns are the properties that you can define for each entity.

If the cluster or group of icons contains more than eight entities, use the scroll bar or arrow keys to go down the list.

The name and SQL statement are defined automatically when entering language mode from database mode.

If a particular entity property has not yet been defined, the cell will be emphasized in a different color, and the Customization Tool will suggest these values:

Class Subclass of thing

CUSTOMIZATION TOOL USER'S GUIDE

<i>Term</i>	Table or column name in database
<i>Category</i>	Noun (after a term has been defined)

If an entity has already been defined, an *o* icon represents a subclass and the *■* icon represents an instance in the class column.

To accept all the suggested values, select *Accept*. You will be given a warning to confirm that you actually do want to accept these values.

To modify an individual name or term, select its cell in the spreadsheet and press *Enter* or select *Fetch*. The contents of the cell are moved to the *Entity data* field at the top of the window, where you can modify the text. To insert it back into the cell, press *Enter* or select *Replace*.

To add a term to the selected term, type it in the *Entity data* field and select *Add*. Alternatively, you can transfer an existing term from another cell using the method for class described below.

To replace or add a subclass or instance to an entity class, perform these steps:

1. Select the cell with the subclass or instance that you want to use.
2. Select *Fetch* to move the value to the *Entity data* field.
3. Select the class cell into which you want to replace or add.
4. Select *Replace* or *Add* as required.

To define the full properties of an individual entity, double click on the property that you want to define or select *Define*.

CUSTOMIZATION TOOL USER'S GUIDE

12 SPECIFYING RELATIONSHIPS

This chapter explains how you specify relationships between entities.

Before specifying relationships, you must:

1. Classify each entity as a subclass
2. Create terms for the entities

12.1 How relationships are shown in the diagram

Relationships are represented by lines in the entity diagram:

- When you go from table mode to entity mode, a dotted line is drawn for each proposed relationship between a table entity and its column entities
- A solid line is drawn between two entities if:
 - You define a relationship between the entities
 - One entity is a subclass or instance of the other
 - One entity consists of several entities, including the other

To change the way relationships are shown in the diagram, select *Relationship icon style* from *Styles*. For example, if you do not want to see the proposed relationships, change the proposed relationship icon to the same color as the screen background.

12.2 Creating or changing a relationship

To define a new relationship or change a relationship that you have already defined:

1. Specify that you want to define a relationship, in either of these ways:
 - Hold down the right mouse button and drag one entity icon onto the other.
 - If a line (dotted or solid) already exists between the two entities, double click on the line. To define a relationship between duster icons, first expand the clusters, and then double click on the relationship line.

The Define relationships window appears.

CUSTOMIZATION TOOL USER'S GUIDE

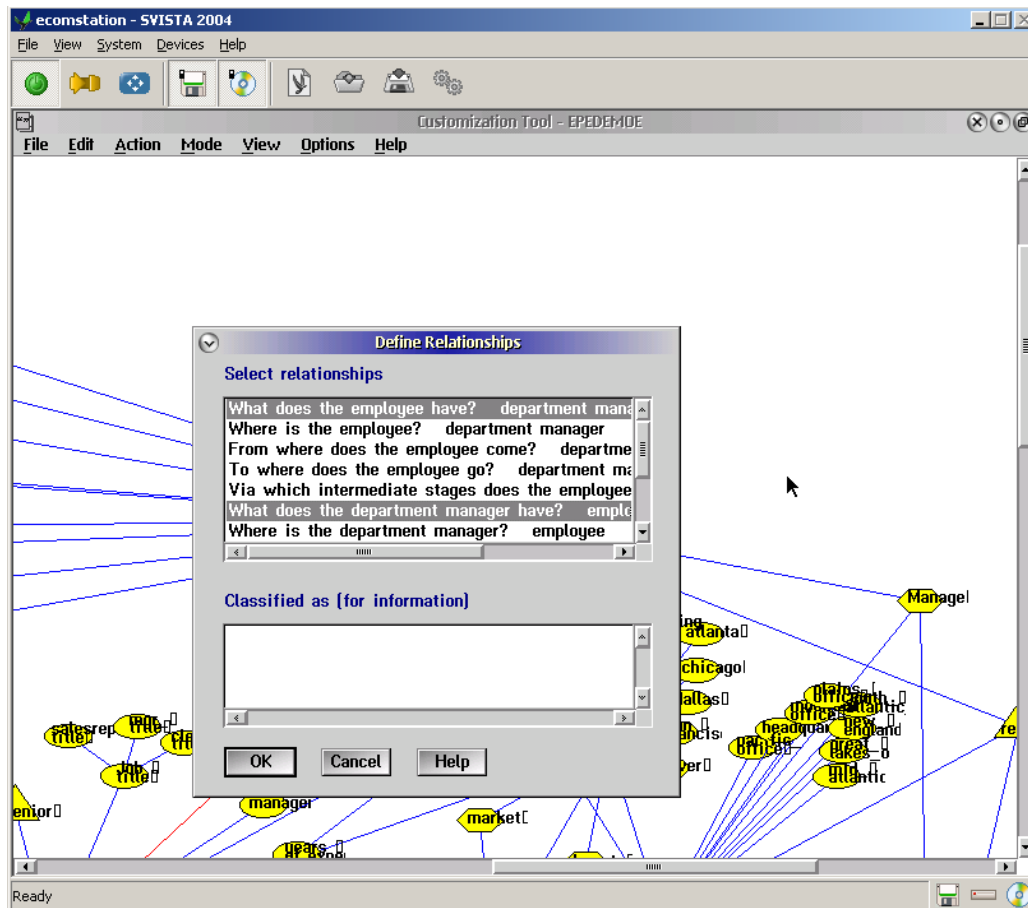


Figure 53. Selecting a relationship.

2. The possible relationships are listed. Select the relationship or relationships that you want.

The possible relationships depend upon the:

- Class of each entity
- Type of terms for each entity (noun, adjective, or verb)
- Syntax of each entity

These sections contain more information about creating relationships:

"Defining name and identifier relationships" on page 36

"Defining refer-to relationships" on page 37

"Possessive relationships" on page 44

"Locative (place) relationships" on page 45

"Temporal (time) relationships" on page 47

"Duration relationship" on page 49

"Measured-by relationship." on page 49

"Prepositions between nouns" on page 51

"Relationships with adjective entities" on page 57

"Associated relationships" on page 64

"Defining verb relationships" on page 70

CUSTOMIZATION TOOL USER'S GUIDE

"Counted-by relationships" on page 72

The *Classified as* box at the bottom of the window indicates if one entity:

- Is a subclass of the other entity
- Is an instance of the other entity
- Consists of several entities, including the other entity

This information is provided for reference only. You cannot change it.

3. Press *OK*. A solid line is drawn between the two entities in the diagram.

12.3 Removing a relationship

To remove a relationship, deselect all of the possible relationships in the Define relationships window.

Note: The entities may still be connected by a dotted line or a solid line, as described in "How relationships are shown in the diagram" on page 130.

12.4 Correcting invalid relationships

Relationships that you previously defined may become invalid if you change the:

- Class of either entity
- Type of terms (noun, adjective, or verb) for either entity
- Syntax of either entity

This is how you check and correct an invalid relationship:

1. Double click on the relationship line to display the *Define relationships* window. A selected but invalid relationship appears within brackets:

[What is the name of department? location]

2. Click once on the invalid relationship to deselect it. The highlighting disappears.

3. Select any of the possible relationships that you want.

4. Press *OK*.

Part 4. Utilities

13 EXTRACTING DATABASE CATALOG INFORMATION

Please contact Dialogue Technologies for more information on this subject.

CUSTOMIZATION TOOL USER'S GUIDE

14 PRODUCING REPORTS

Please contact Dialogue Technologies for more information on this subject.

Part 5. Appendices

Appendix A. SQL syntax in the Customization Tool

This chapter describes the form of SQL statements that may be used to specify the database representation of entities in the Customization Tool. The natural language engine uses these statements to generate the SQL to be sent to the database. In doing this, the engine uses a far greater subset of the SQL language than that supported by the Customization Tool.

General constraints

The syntax of SQL statements that can be used within Ergo entities has these general restrictions:

The SQL for a Ergo entity may only consist of a single SELECT statement. The UNION operator is not permitted.

The SELECT statement can return only a single column or a single expression. Consider creation of composite entities if more than one column is desired. See "Composite entities" on page 61.

The names of tables or views must be specified with the name of the creator and with a correlation name.

A column is specified by its correlation name and column name. Ergo uses this for SQL optimization.

The DISTINCT and ALL clauses are not permitted in the SELECT clause.

Scalar functions, DECIMAL, FLOAT, INTEGER, LENGTH, SUBSTR, VALUE, and VARGRAPHIC, are not permitted.

Special registers, CURRENT SERVER and CURRENT TIMEZONE, are not permitted.

The CONCAT operator is not permitted in expressions.

Host variables may not be used.

GROUP BY, HAVING, ORDER BY or FOR UPDATE OF clauses are not permitted.

An example of a valid Ergo entity SQL statement is:

```
SELECT X1.ID FROM EPE.STAFF X1 WHERE X1.JOB = 'CLERK'
```

How to read the syntax diagrams

Throughout this chapter, syntax is described as follows:

Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

CUSTOMIZATION TOOL USER'S GUIDE

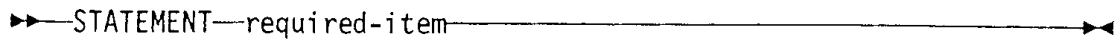
The \blacktriangleright — symbol indicates the beginning of a statement.

The — \blacktriangleright symbol indicates that the statement syntax is continued on the next line.

The \blacktriangleright — symbol indicates that a statement is continued from a previous line.

The — \blacktriangleright \blacktriangleleft symbol indicates the end of a statement.

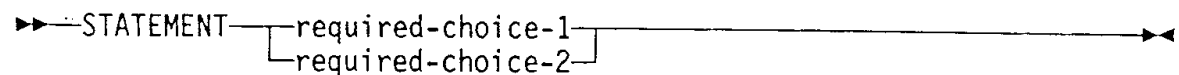
- Required items appear on the horizontal line (main path).



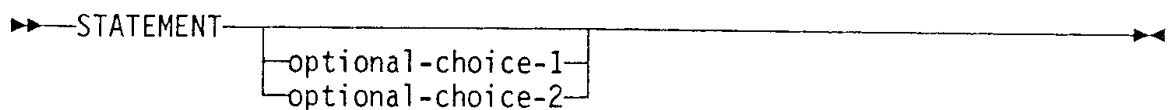
- Optional items appear below the main path.



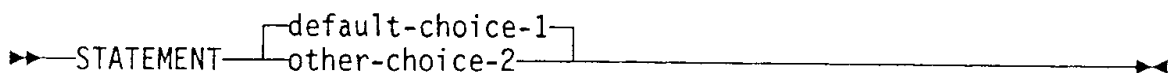
- If you can choose from two or more items, they appear vertically, in a stack. If you must choose one of the items, one item of the stack appears on the main path.



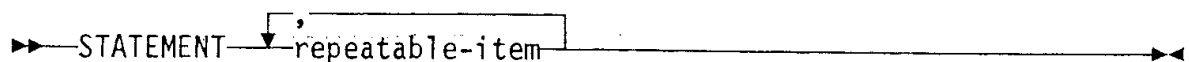
- If choosing one of the items is optional, the entire stack appears below the main path.



- Parameters that are above the main path are default parameters.



- An arrow returning to the left above the item indicates an item that you can repeat. Required items appear on the main line and optional items appear below the main line.



A repeat arrow indicates that you can make more than one choice from the stacked items, or repeat a single item. If a separator is required between items, it is shown in the repeat arrow.

- Keywords appear in upper case (for example, STATEMENT).
- References to other syntax definitions appear in lower case (for example, *expression*).
- References to variables appear in italics.
- Parentheses and commas must be entered as part of the command syntax as shown.

CUSTOMIZATION TOOL USER'S GUIDE

SQL syntax diagrams

This section gives a detailed description of the syntax of SQL statements allowed in Ergo entities.

Select statement

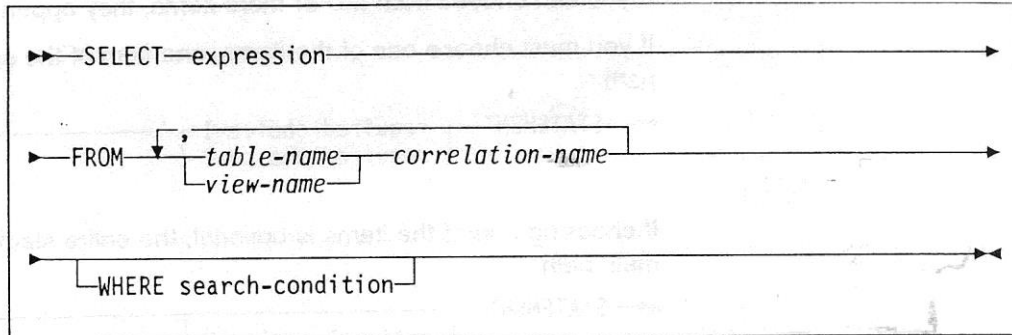


Figure 54 Select statement syntax

Search-conditions

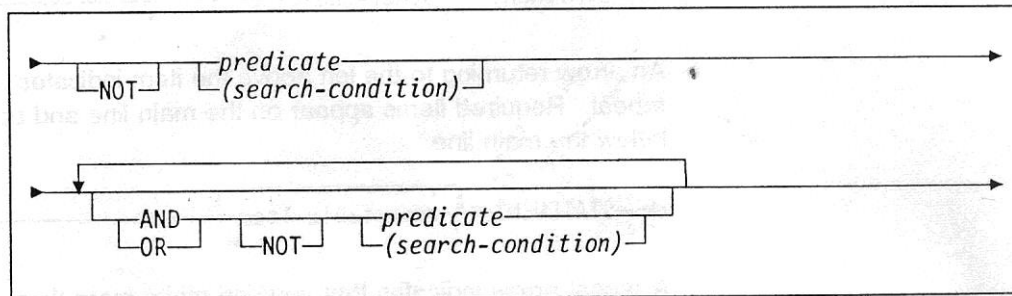


Figure 55 Search-condition syntax

Expressions

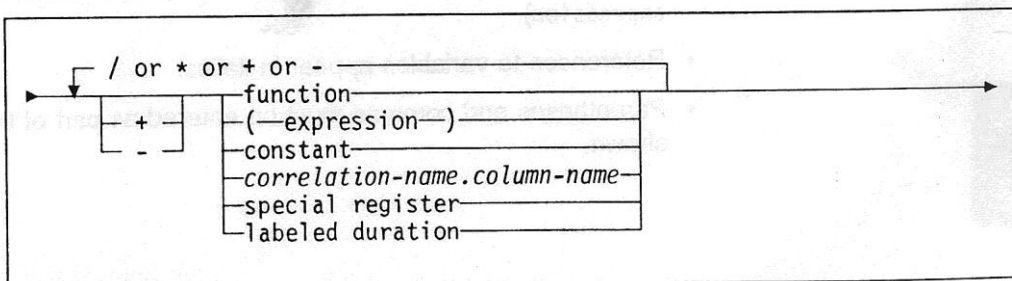


Figure 56 Expression syntax

Expressions in SQL for Ergo entities differ from the standard syntax in these ways:

- Host variables cannot be used.
- Column names must include the correlation name.

CUSTOMIZATION TOOL USER'S GUIDE

Column functions

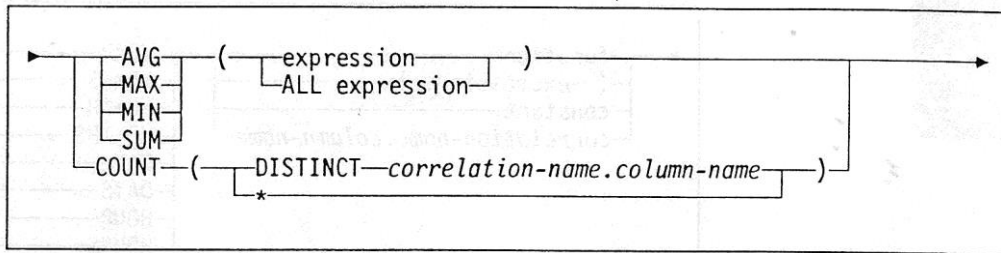


Figure 57 Column function syntax

Column functions in SQL for Ergo entities differ from the standard syntax in this way:

- Column names must include the correlation name.

Scalar functions

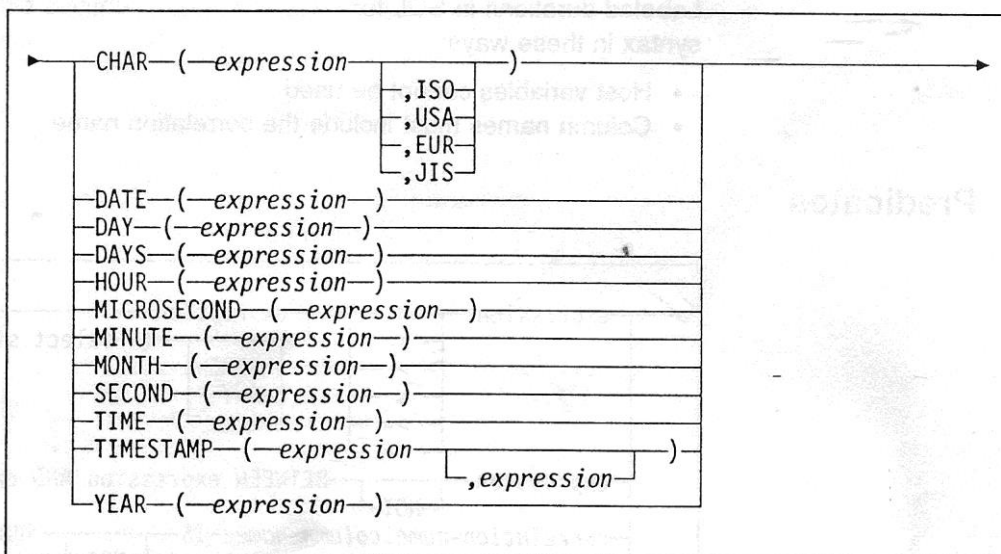


Figure 58 Scalar function syntax

Special registers

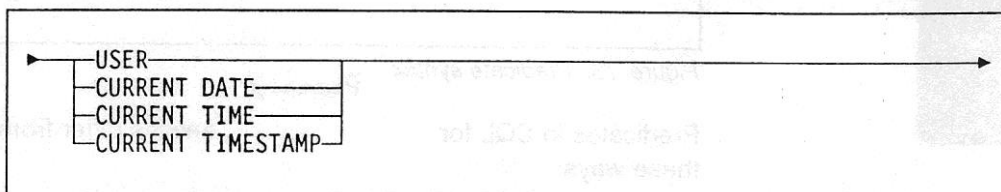


Figure 59 Special register syntax

Labeled durations

CUSTOMIZATION TOOL USER'S GUIDE

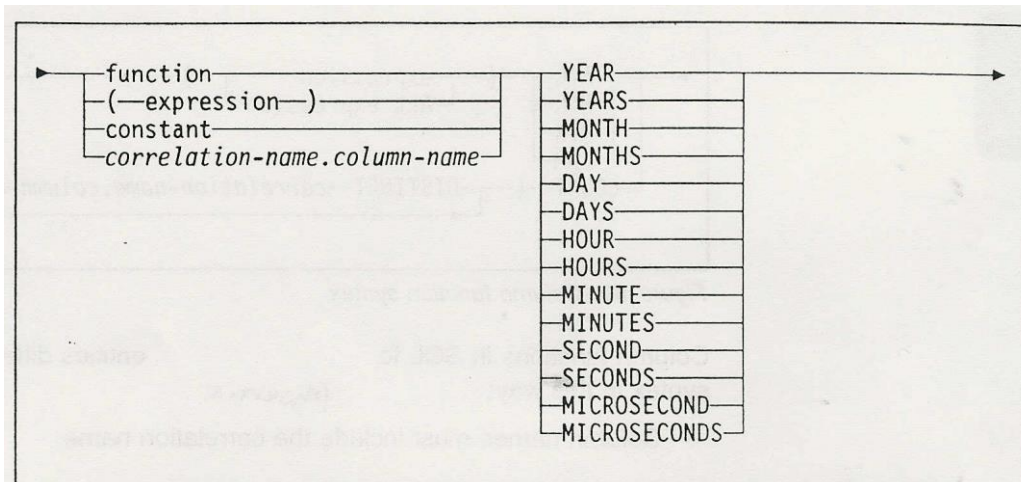


Figure 60 Labeled duration syntax

Labeled durations in SQL for Ergo entities differ from the standard syntax in these ways:

- Host variables cannot be used.
- Column names must include the correlation name.

Predicates

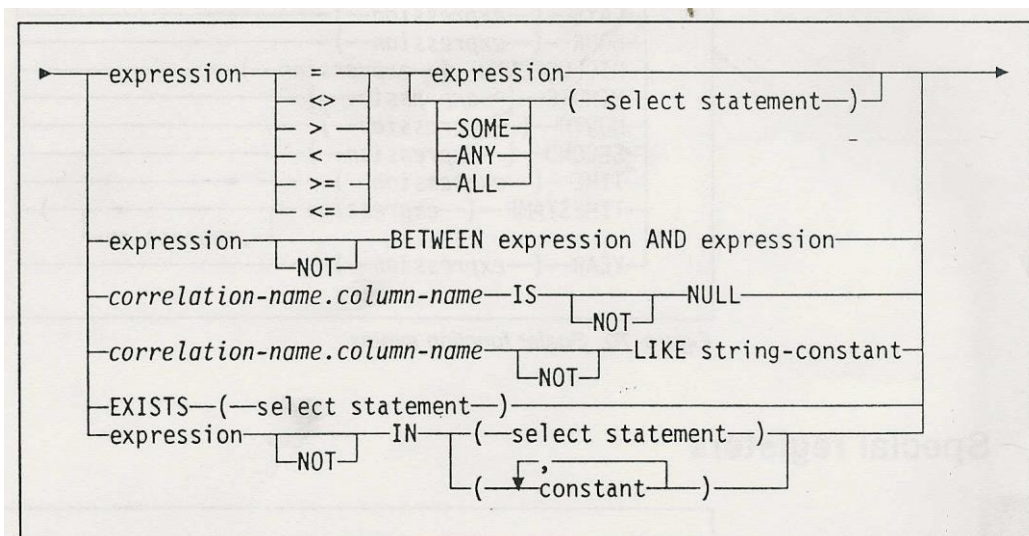


Figure 61 Predicate syntax

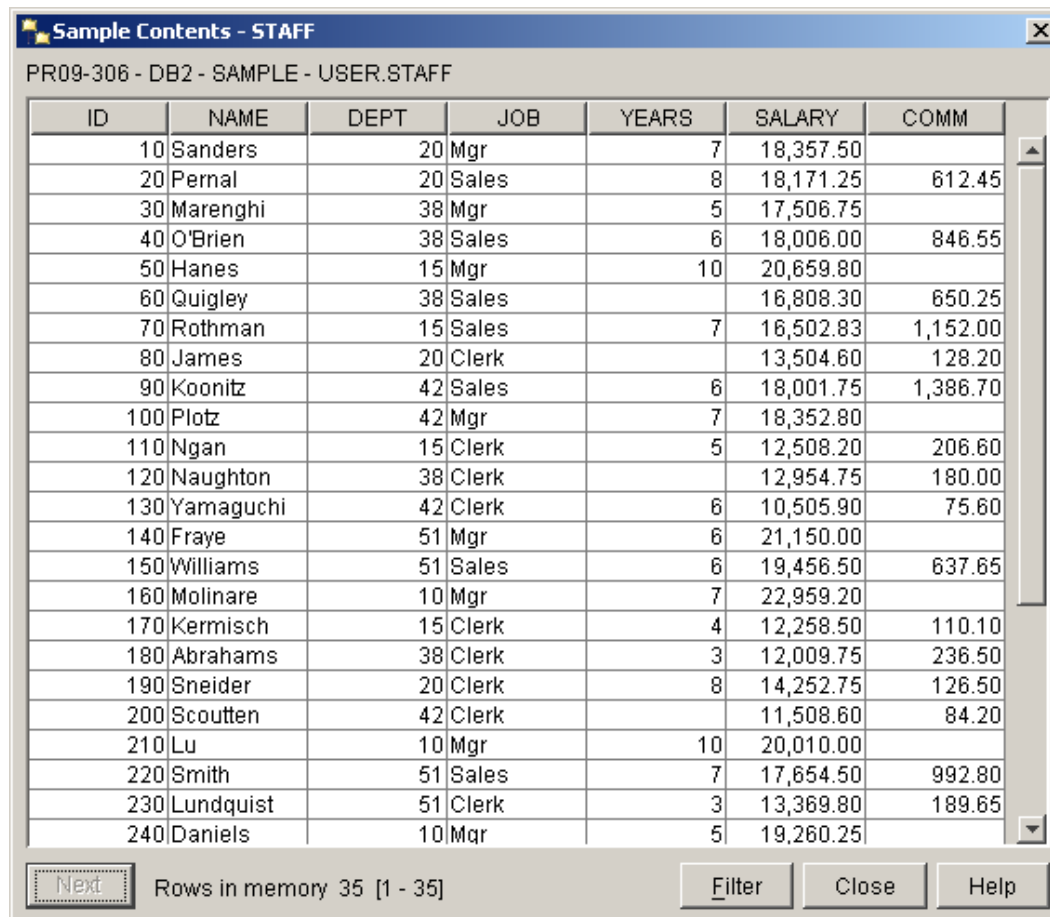
Predicates in SQL for Ergo entities differ from the standard syntax in these ways:

- The \neq , \rightarrow , and \rightarrow operators may not be used.
- Host variables are not valid in the LIKE or IN predicates.
- Column names must include the correlation name.

Appendix B. Sample database used in this document.

The examples used throughout this document represent a data mining application, where users can ask questions about a fictitious company by addressing the database with natural language queries.

The database is a sample database shipped with IBM DB2 and contains tables with information about products, projects, sales, staff, office locations, etc.



ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
10	Sanders	20	Mgr	7	18,357.50	
20	Pernal	20	Sales	8	18,171.25	612.45
30	Marenghi	38	Mgr	5	17,506.75	
40	O'Brien	38	Sales	6	18,006.00	846.55
50	Hanes	15	Mgr	10	20,659.80	
60	Quigley	38	Sales		16,808.30	650.25
70	Rothman	15	Sales	7	16,502.83	1,152.00
80	James	20	Clerk		13,504.60	128.20
90	Koonitz	42	Sales	6	18,001.75	1,386.70
100	Plotz	42	Mgr	7	18,352.80	
110	Ngan	15	Clerk	5	12,508.20	206.60
120	Naughton	38	Clerk		12,954.75	180.00
130	Yamaguchi	42	Clerk	6	10,505.90	75.60
140	Fraye	51	Mgr	6	21,150.00	
150	Williams	51	Sales	6	19,456.50	637.65
160	Molinare	10	Mgr	7	22,959.20	
170	Kermisch	15	Clerk	4	12,258.50	110.10
180	Abrahams	38	Clerk	3	12,009.75	236.50
190	Sneider	20	Clerk	8	14,252.75	126.50
200	Scoutten	42	Clerk		11,508.60	84.20
210	Lu	10	Mgr	10	20,010.00	
220	Smith	51	Sales	7	17,654.50	992.80
230	Lundquist	51	Clerk	3	13,369.80	189.65
240	Daniels	10	Mgr	5	19,260.25	

Figure 62. Staff table in the sample database.

CUSTOMIZATION TOOL USER'S GUIDE

EMPNO	FIRSTNAME	MIDINIT	LASTNAME	WORKDE...	PHONENO	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDATE	SALARY	BONUS	COMM
000010	CHRISTINE	I	HAAS	A00	3978	Jan 1, 1965	PRES		18 F	Aug 24, 1933	52,750.00	1,000.00	4,220.00
000020	MICHAEL	L	THOMPSON	B01	3476	Oct 10, 1973	MANAGER		18 M	Feb 2, 1948	41,250.00	800.00	3,300.00
000030	SALLY	A	KWAN	C01	4738	Apr 5, 1975	MANAGER		20 F	May 11, 1941	38,250.00	800.00	3,060.00
000050	JOHN	B	GEYER	E01	6789	Aug 17, 1949	MANAGER		16 M	Sep 15, 19...	40,175.00	800.00	3,214.00
000060	IRVING	F	STERN	D11	6423	Sep 14, 19...	MANAGER		16 M	Jul 7, 1945	32,250.00	500.00	2,580.00
000070	EVA	D	PULASKI	D21	7831	Sep 30, 19...	MANAGER		16 F	May 26, 1953	36,170.00	700.00	2,893.00
000090	EILEEN	W	HENDERS...	E11	5498	Aug 15, 1970	MANAGER		16 F	May 15, 1941	29,750.00	600.00	2,380.00
000100	THEODORE	Q	SPENSER	E21	0972	Jun 19, 1980	MANAGER		14 M	Dec 18, 19...	26,150.00	500.00	2,092.00
000110	VINCENZO	G	LUCCHESSI	A00	3490	May 16, 1958	SALESREP		19 M	Nov 5, 1929	46,500.00	900.00	3,720.00
000120	SEAN		O'CONNELL	A00	2167	Dec 5, 1963	CLERK		14 M	Oct 18, 1942	29,250.00	600.00	2,340.00
000130	DOLORES	M	QUINTANA	C01	4578	Jul 28, 1971	ANALYST		16 F	Sep 15, 19...	23,800.00	500.00	1,904.00
000140	HEATHER	A	NICHOLLS	C01	1793	Dec 15, 19...	ANALYST		18 F	Jan 19, 1946	28,420.00	600.00	2,274.00
000150	BRUCE		ADAMSON	D11	4510	Feb 12, 1972	DESIGNER		16 M	May 17, 1947	25,280.00	500.00	2,022.00
000160	ELIZABETH	R	PIANKA	D11	3782	Oct 11, 1977	DESIGNER		17 F	Apr 12, 1955	22,250.00	400.00	1,780.00
000170	MASATOSHI	J	YOSHIMURA	D11	2890	Sep 15, 19...	DESIGNER		16 M	Jan 5, 1951	24,680.00	500.00	1,974.00
000180	MARILYN	S	SCOUTTEN	D11	1682	Jul 7, 1973	DESIGNER		17 F	Feb 21, 1949	21,340.00	500.00	1,707.00
000190	JAMES	H	WALKER	D11	2986	Jul 26, 1974	DESIGNER		16 M	Jun 25, 1952	20,450.00	400.00	1,636.00
000200	DAVID		BROWN	D11	4501	Mar 3, 1966	DESIGNER		16 M	May 29, 1941	27,740.00	600.00	2,217.00
000210	WILLIAM	T	JONES	D11	0942	Apr 11, 1979	DESIGNER		17 M	Feb 23, 1953	18,270.00	400.00	1,462.00
000220	JENNIFER	K	LUTZ	D11	0672	Aug 29, 1968	DESIGNER		18 F	Mar 19, 1948	29,840.00	600.00	2,387.00
000230	JAMES	J	JEFFERSON	D21	2094	Nov 21, 1966	CLERK		14 M	May 30, 1935	22,180.00	400.00	1,774.00
000240	SALVATORE	M	MARINO	D21	3780	Dec 5, 1979	CLERK		17 M	Mar 31, 1954	28,760.00	600.00	2,301.00
000250	DANIEL	S	SMITH	D21	0961	Oct 30, 1969	CLERK		15 M	Nov 12, 1939	19,180.00	400.00	1,534.00

Figure 63. Table with detailed employee information in the sample database.

SALES_D_...	SALES_P...	REGION	SALES
Dec 31, 19...	LUCCHESSI	Ontario-So...	1
Dec 31, 19...	LEE	Ontario-So...	3
Dec 31, 19...	LEE	Quebec	1
Dec 31, 19...	LEE	Manitoba	2
Dec 31, 19...	GOUNOT	Quebec	1
Mar 29, 1996	LUCCHESSI	Ontario-So...	3
Mar 29, 1996	LUCCHESSI	Quebec	1
Mar 29, 1996	LEE	Ontario-So...	2
Mar 29, 1996	LEE	Ontario-No...	2
Mar 29, 1996	LEE	Quebec	3
Mar 29, 1996	LEE	Manitoba	5
Mar 29, 1996	GOUNOT	Ontario-So...	3
Mar 29, 1996	GOUNOT	Quebec	1
Mar 29, 1996	GOUNOT	Manitoba	7
Mar 30, 1996	LUCCHESSI	Ontario-So...	1
Mar 30, 1996	LUCCHESSI	Quebec	2
Mar 30, 1996	LUCCHESSI	Manitoba	1
Mar 30, 1996	LEE	Ontario-So...	7
Mar 30, 1996	LEE	Ontario-No...	3
Mar 30, 1996	LEE	Quebec	7
Mar 30, 1996	LEE	Manitoba	4
Mar 30, 1996	GOUNOT	Ontario-So...	2
Mar 30, 1996	GOUNOT	Quebec	18
Mar 30, 1996	GOUNOT	Manitoba	1

Figure 64. Information about sales figures in the company.

CUSTOMIZATION TOOL USER'S GUIDE

PR09-306 - DB2 - SAMPLE - USER.PROJECT

PROJNO	PROJNAME	DEPTNO	RESPEMP	PRSTAFF	PRSTDATE	PRENDATE	MAJPROJ
AD3100	ADMIN SE...	D01	000010	6.50	Jan 1, 1982	Feb 1, 1983	
AD3110	GENERAL ...	D21	000070	6.00	Jan 1, 1982	Feb 1, 1983	AD3100
AD3111	PAYROLL ...	D21	000230	2.00	Jan 1, 1982	Feb 1, 1983	AD3110
AD3112	PERSONN...	D21	000250	1.00	Jan 1, 1982	Feb 1, 1983	AD3110
AD3113	ACCOUNT ...	D21	000270	2.00	Jan 1, 1982	Feb 1, 1983	AD3110
IF1000	QUERY SE...	C01	000030	2.00	Jan 1, 1982	Feb 1, 1983	
IF2000	USER EDU...	C01	000030	1.00	Jan 1, 1982	Feb 1, 1983	
MA2100	WELD LIN...	D01	000010	12.00	Jan 1, 1982	Feb 1, 1983	
MA2110	W L PROG...	D11	000060	9.00	Jan 1, 1982	Feb 1, 1983	MA2100
MA2111	W L PROG...	D11	000220	2.00	Jan 1, 1982	Dec 1, 1982	MA2110
MA2112	W L ROBO...	D11	000150	3.00	Jan 1, 1982	Dec 1, 1982	MA2110
MA2113	W L PROD ...	D11	000160	3.00	Feb 15, 1982	Dec 1, 1982	MA2110
OP1000	OPERATIO...	E01	000050	6.00	Jan 1, 1982	Feb 1, 1983	
OP1010	OPERATION	E11	000090	5.00	Jan 1, 1982	Feb 1, 1983	OP1000
OP2000	GEN SYST...	E01	000050	5.00	Jan 1, 1982	Feb 1, 1983	
OP2010	SYSTEMS ...	E21	000100	4.00	Jan 1, 1982	Feb 1, 1983	OP2000
OP2011	SCP SYST...	E21	000320	1.00	Jan 1, 1982	Feb 1, 1983	OP2010
OP2012	APPLICATI...	E21	000330	1.00	Jan 1, 1982	Feb 1, 1983	OP2010
OP2013	DB/DC SU...	E21	000340	1.00	Jan 1, 1982	Feb 1, 1983	OP2010
PL2100	WELD LIN...	B01	000020	1.00	Jan 1, 1982	Sep 15, 19...	MA2100

Next Rows in memory 20 [1 - 20] Filter Close Help

Figure 65. Table containing information about development projects.

PR09-306 - DB2 - SAMPLE - USER.ORG

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
10	Head Office	160	Corporate	New York
15	New Engla...	50	Eastern	Boston
20	Mid Atlantic	10	Eastern	Washington
38	South Atlan...	30	Eastern	Atlanta
42	Great Lakes	100	Midwest	Chicago
51	Plains	140	Midwest	Dallas
66	Pacific	270	Western	San Franci...
84	Mountain	290	Western	Denver

Next Rows in memory 8... Filter Close Help

Figure 66. Table with company organization.