Getting Started with ER*win*

Introduction

ER*win* is a popular data modeling tool used by a number of major companies in Omaha and throughout the world. The product is currently owned, developed, and marketed by Computer Associates, a leading software developer. The product supports a variety of aspects of database design, including data modeling, forward engineering (the creation of a database schema and physical database on the basis of a data model), and reverse engineering (the creation of a data model on the basis of an existing database) for a wide variety of relational DBMS, including Microsoft Access, Oracle, DB2, Sybase, and others.

This brief tutorial steps you through the process of creating a data model using ER*win*.   It will not explain all aspects of ERwin, but will show you the minimum necessary to create and use data models for this class. It consists of three major segments, which correspond to the project-related assignments in your class:

1. Creation of a basic data model (Conceptual data model)
2. Creation of a database schema
3. Creation of the database

This tutorial is a static one, suitable for printing. A tutorial using screen captures and narration is also available.

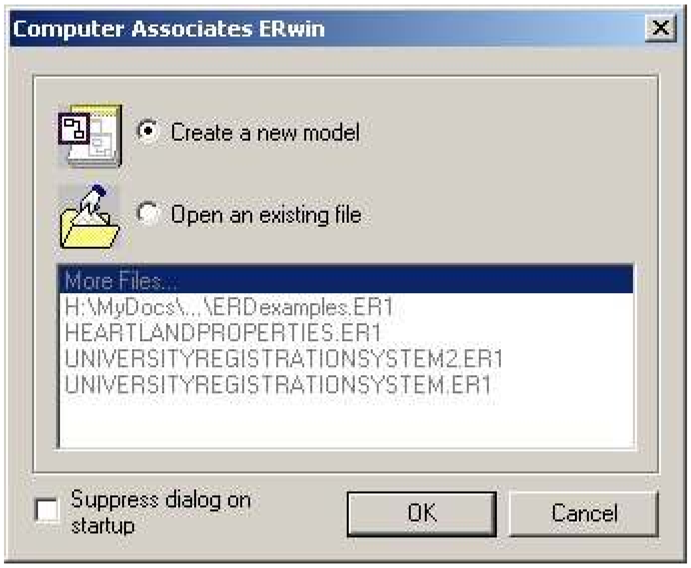
Section 1. Creation of a basic data model

You will be creating a data model similar to that created for the Heartland Properties case study used in class. The Entities involved in this model include: Employee, Office, Property, and Inspection.

First, invoke ER*win* from Start->Programs->Computer Associates->All Fusion-> ERwin Data Modeler-> ERwin

You will first encounter a dialog box entitled "ModelMart Connection Manager". Simply click on Cancel.

You will be asked to choose between creating a new model, and opening an existing model, as shown in Figure 1. For this exercise, create a new model.



Computer Associates ERwin

[✓ Create a new model]

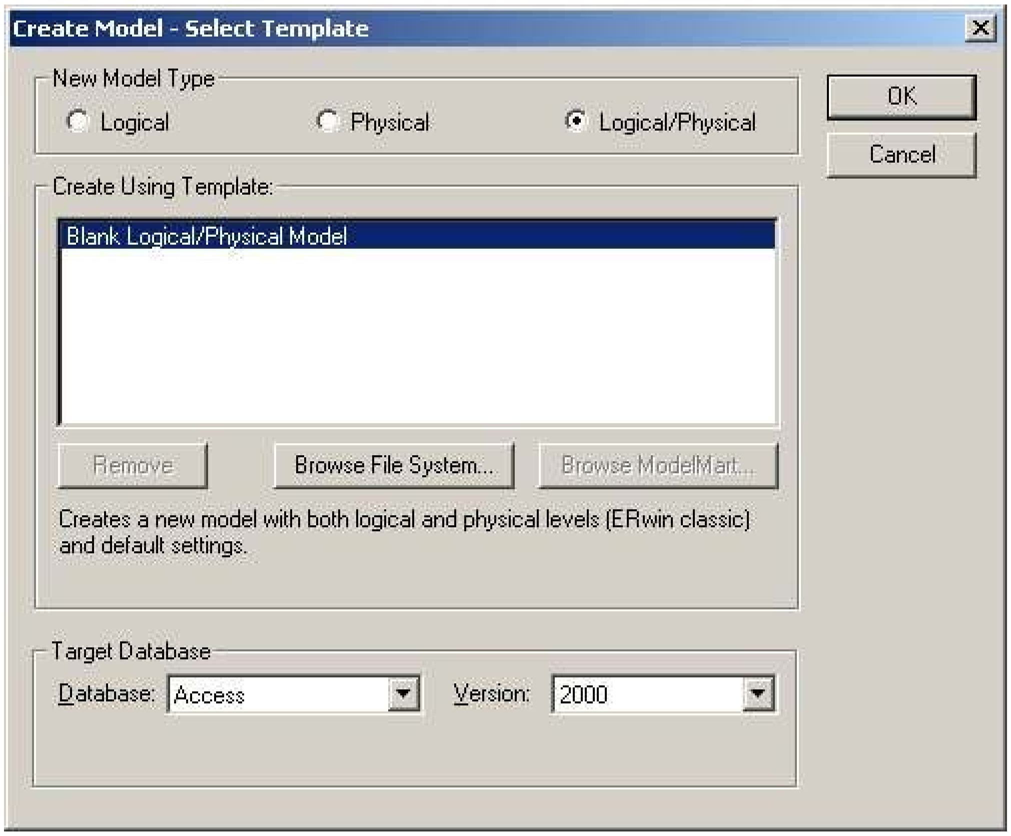
Open an existing file

More Files...
H:\MyDocs\...\ERDexamples.ER1
HEARTLANDPROPERTIES.ER1
UNIVERSITYREGISTRATIONSYSTEM2.ER1
UNIVERSITYREGISTRATIONSYSTEM.ER1

Suppress dialog on startup

Figure 1: Create a new model

The next dialog box, shown in Figure 2, will ask you to choose the template to be used to create the new model. At this point, you may associate the model with a target DBMS product, such as Microsoft Access, Oracle, DB2, etc. You will have the opportunity to change this choice later, but for now choose Microsoft Access. Also, choose Logical/Physical as the new model type. This choice will allow us to switch back and forth easily between a logical model (ER Diagram) and a physical model (database schema).



Create Model - Select Template

New Model Type: Logical Physical [✓ Logical/Physical]

Create Using Template:

Blank Logical/Physical Model

Creates a new model with both logical and physical levels (ERwin classic) and default settings.

Target Database

Database: Access Version: 2000

Figure 2: Selecting a model type

ERwin will now display the main window from which most of your ER diagram development will be done, as shown in Figure 3.

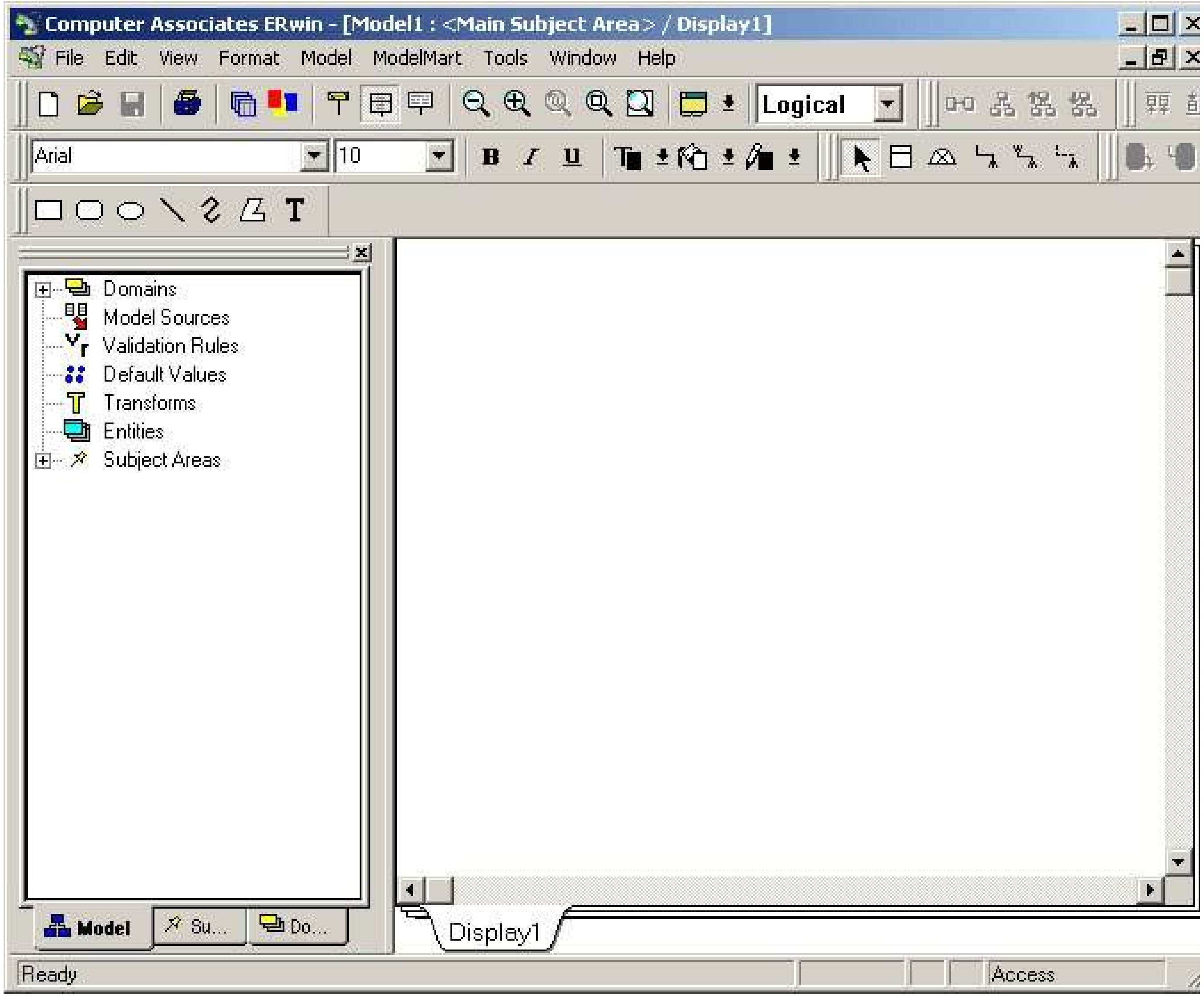


Figure 3: The ERwin Workplace

The ERwin workplace consists of two main parts. On the left is the Model Navigator, which displays a hierarchy of items of importance, such as entities, domains, and subject areas. On the right is the Display Window, which will show the ER diagram itself. As you create objects, they will appear in the display window (if they are visual in nature, like entities), and appear in the hierarchy within the Model Navigator.

Setting Preferences

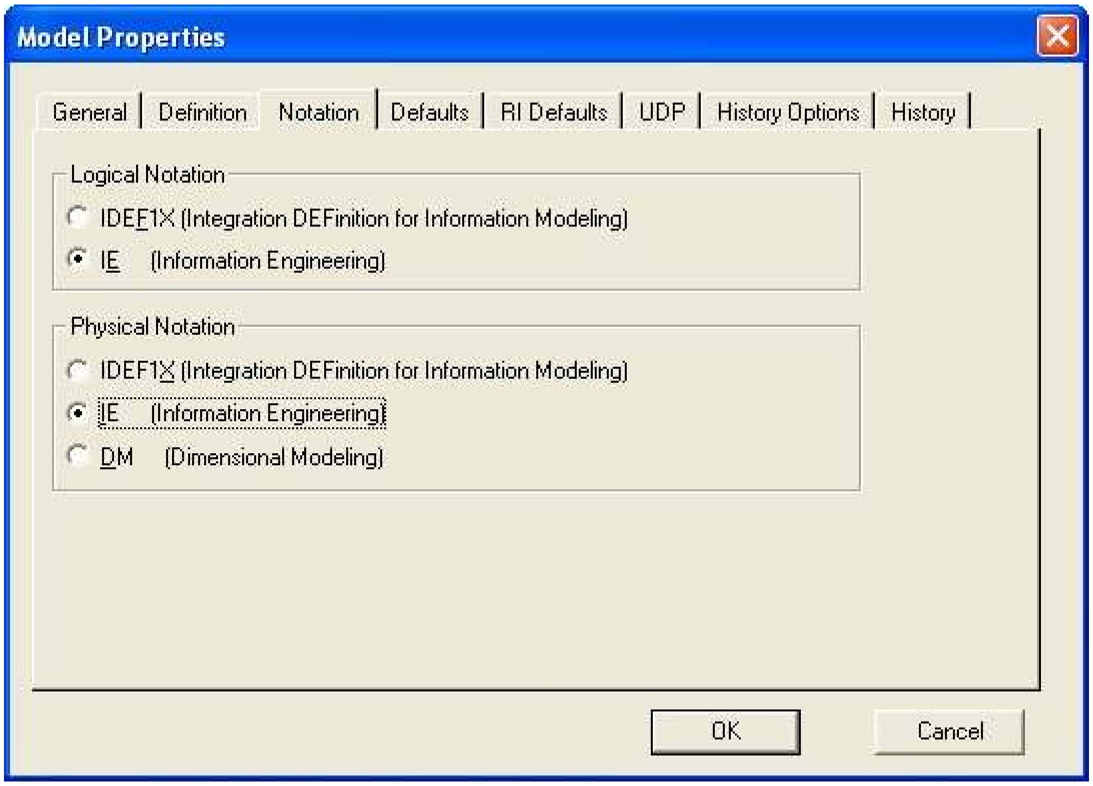
A tool like ER*win* can accomodate a number of data modeling notations and conventions. In my class please make the following changes to the preferences before you begin to create your model:

1. From the Format->Entity Display menu, make sure that **Primary Key Designator** is ***checked***
2. From the Format->Entity Display menu, make sure that **Foreign Key Designator (FK)** is

unchecked.

3. From the Format->Entity Display menu, make sure that **Show Migrated Attributes** is ***unchecked***.
4. From the Format->Relationship Display, make sure that **Verb Phrase** is ***checked***.

In addition, you may choose between two different E-R diagramming notations. In ERwin, click on Model->Model Properties to see the window shown in Figure 3b:



Logical Notation
- ( ) IDEF1X (Integration DEFinition for Information Modeling)
- (●) IE (Information Engineering)

Physical Notation
- ( ) IDEF1X (Integration DEFinition for Information Modeling)
- (●) IE (Information Engineering)
- ( ) DM (Dimensional Modeling)

Figure 3b: Choice of notation

This tutorial is based on the Information Engineering ("Crows Foot") notation. You may also use IDEF1X, if you wish.

Creating an Entity

To create a new entity, click on the entity icon ( ) on the toolbar, or right-click on the word *Entity* in the Model Navigator. If you click on the entity icon, you then should click on the Display Window where you would like the entity to appear, as shown in Figure 4.

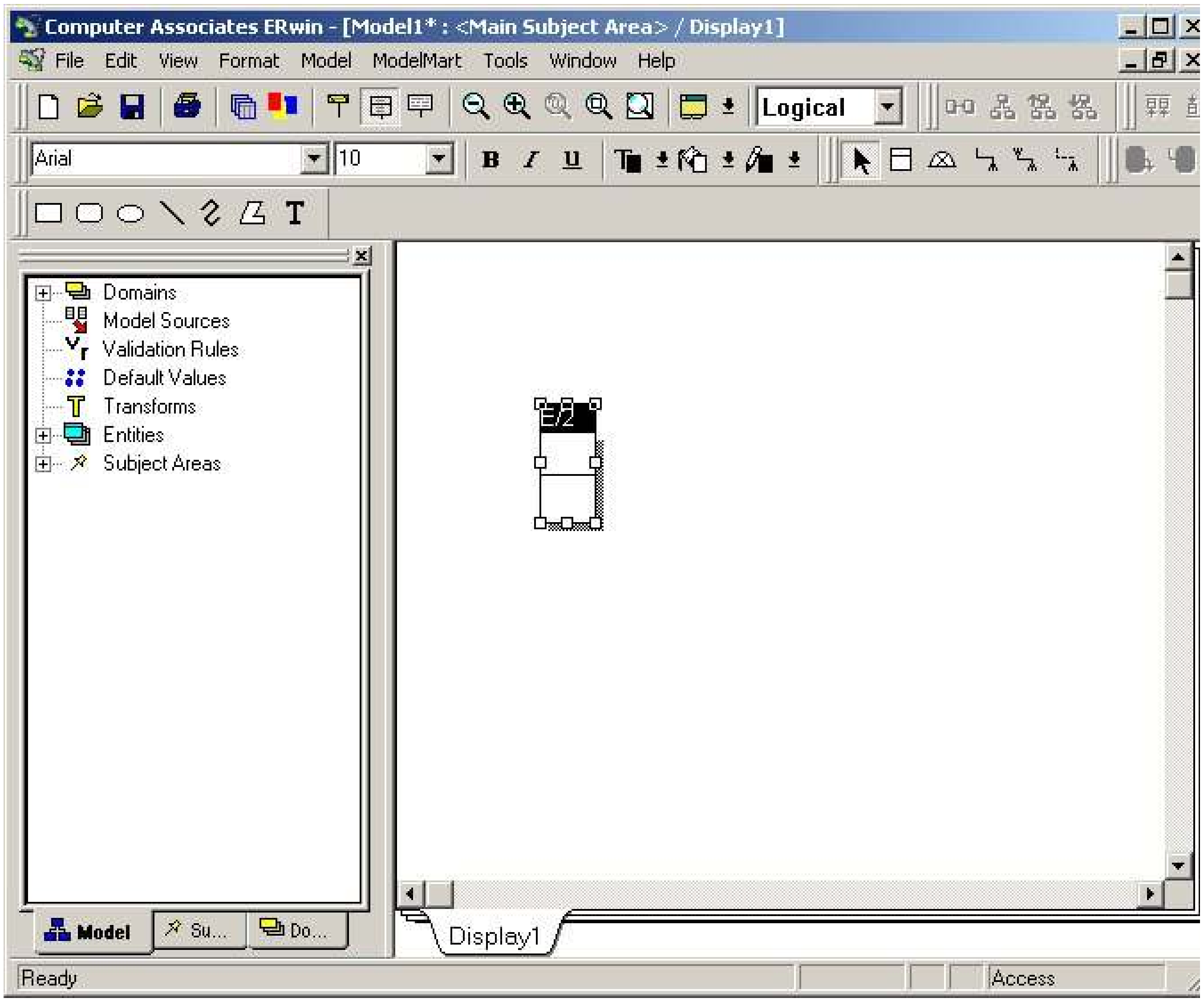


Figure 4: A new entity

Notice that the default name for the entity is E/x, where x is some number (2 in this case). Click on the *Tab* key several times and notice what happens. Pressing the *tab* key cause the focus to cycle between the three main parts of the Entity: the name of the entity, the primary key attribute(s), and the non-primary key attribute(s). In general, to modify one of these three parts of the entity, you will press the *Tab* key to cycle to the appropriate part of the entity, then type to add or modify that part of the entity.

Right now, press the *Tab* key until the entity name is highlighted. Then type EMPLOYEE, as shown in Figure 5.

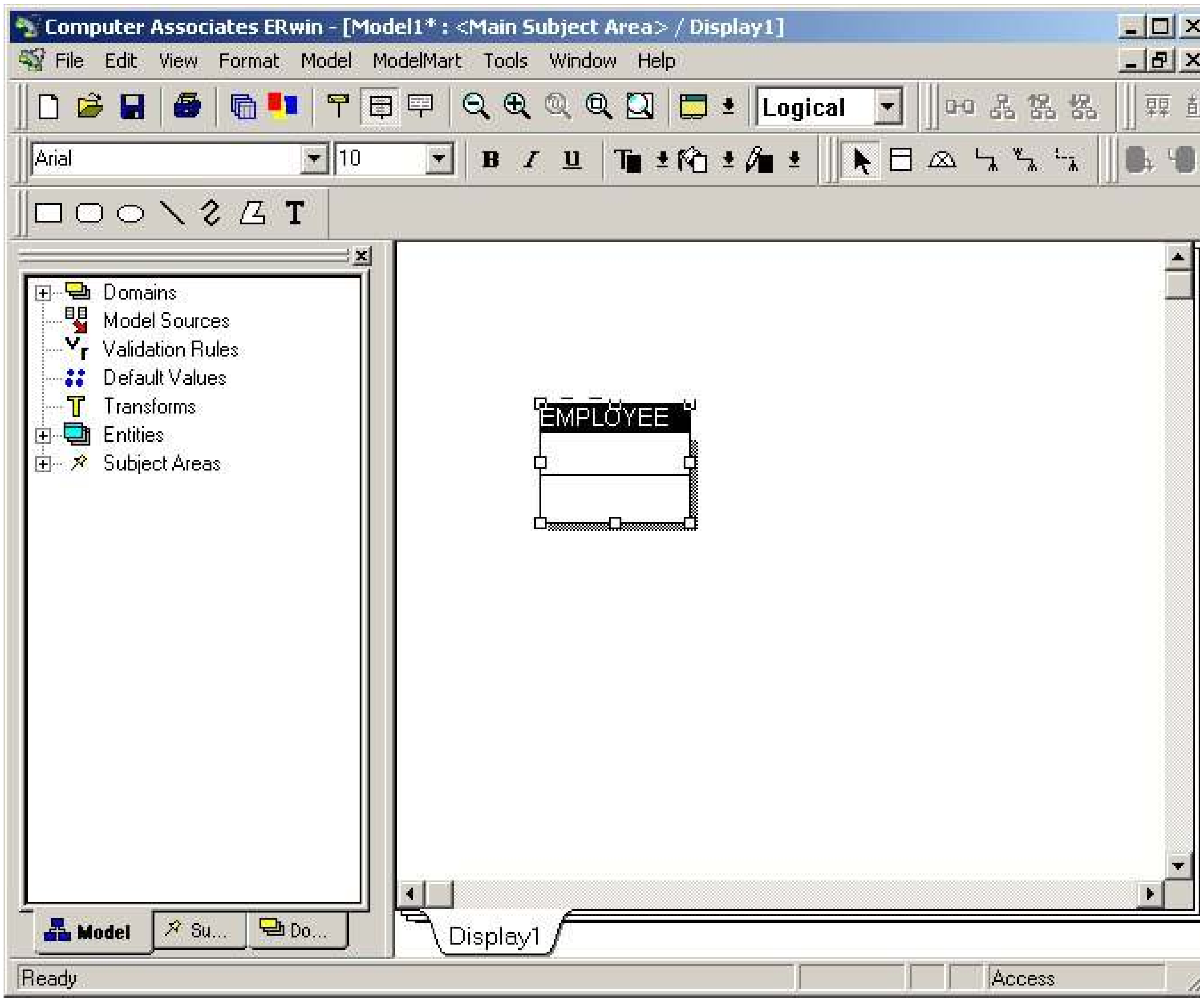


Figure 5: Changing the name of the entity

At this point, you may wish to save and name your diagram to avoid loss should the system or application crash.

Adding primary key columns

Once you have changed the name to EMPLOYEE, press the *Tab* key again to move the focus to the next part of the Entity, adding a primary key attribute. Then type the name of the primary key attribute, Emp_Num, as shown in Figure 6.

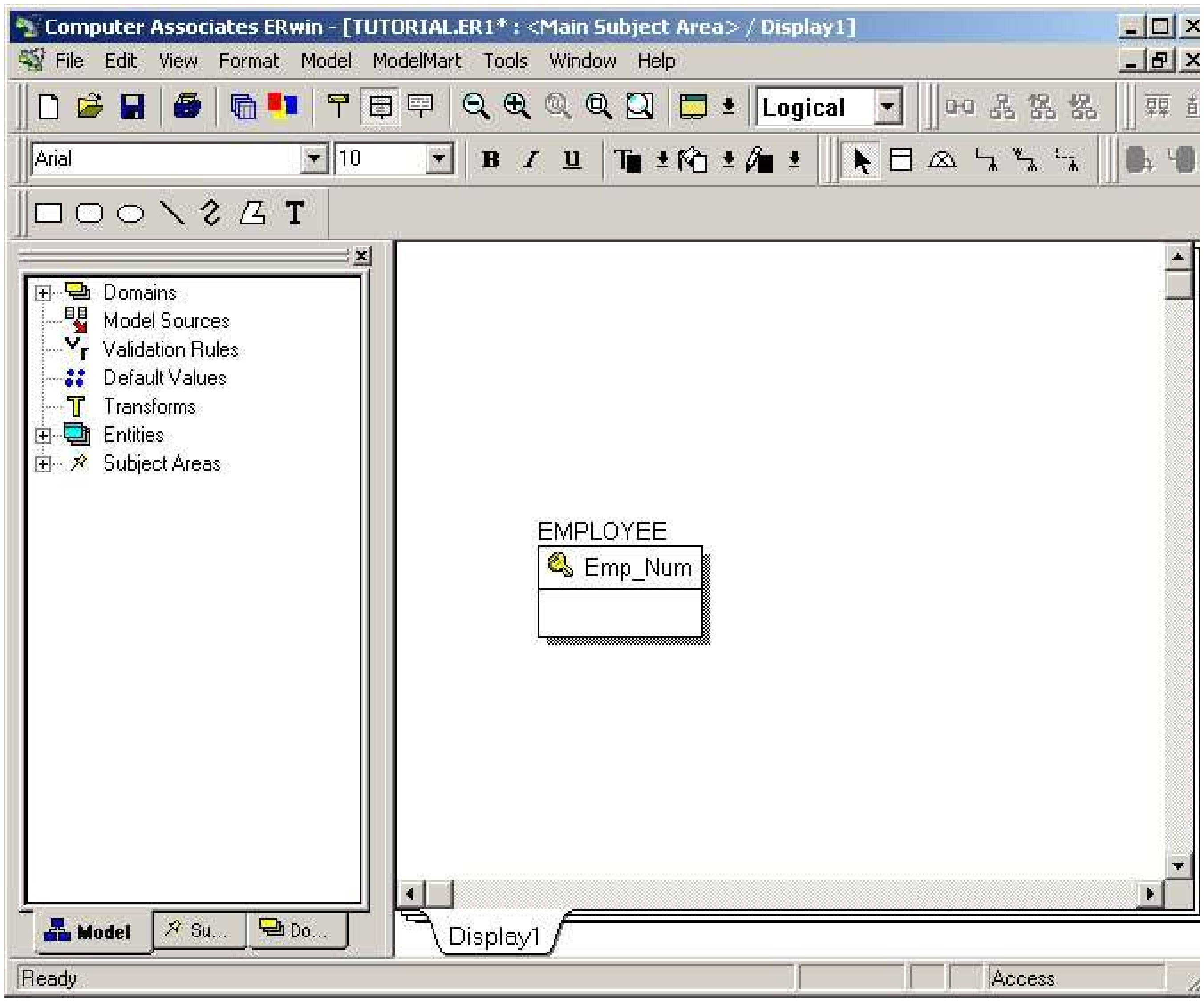


Figure 6: Adding a primary key attribute

Notice that because of the preferences you set earlier, the primary key attribute has a key icon next to it.

Press the *Tab* key one more time to bring the focus below the horizontal line in the Entity, where you will add in a number of non-primary key attributes. Type Emp_Fname, as shown in Figure 7. When you have typed Emp_Name, press the *Enter* key (not *Tab*). Notice what happens. The cursor is now positioned for you to add another attribute in this same portion of the Entity, the non-primary key attribute portion.

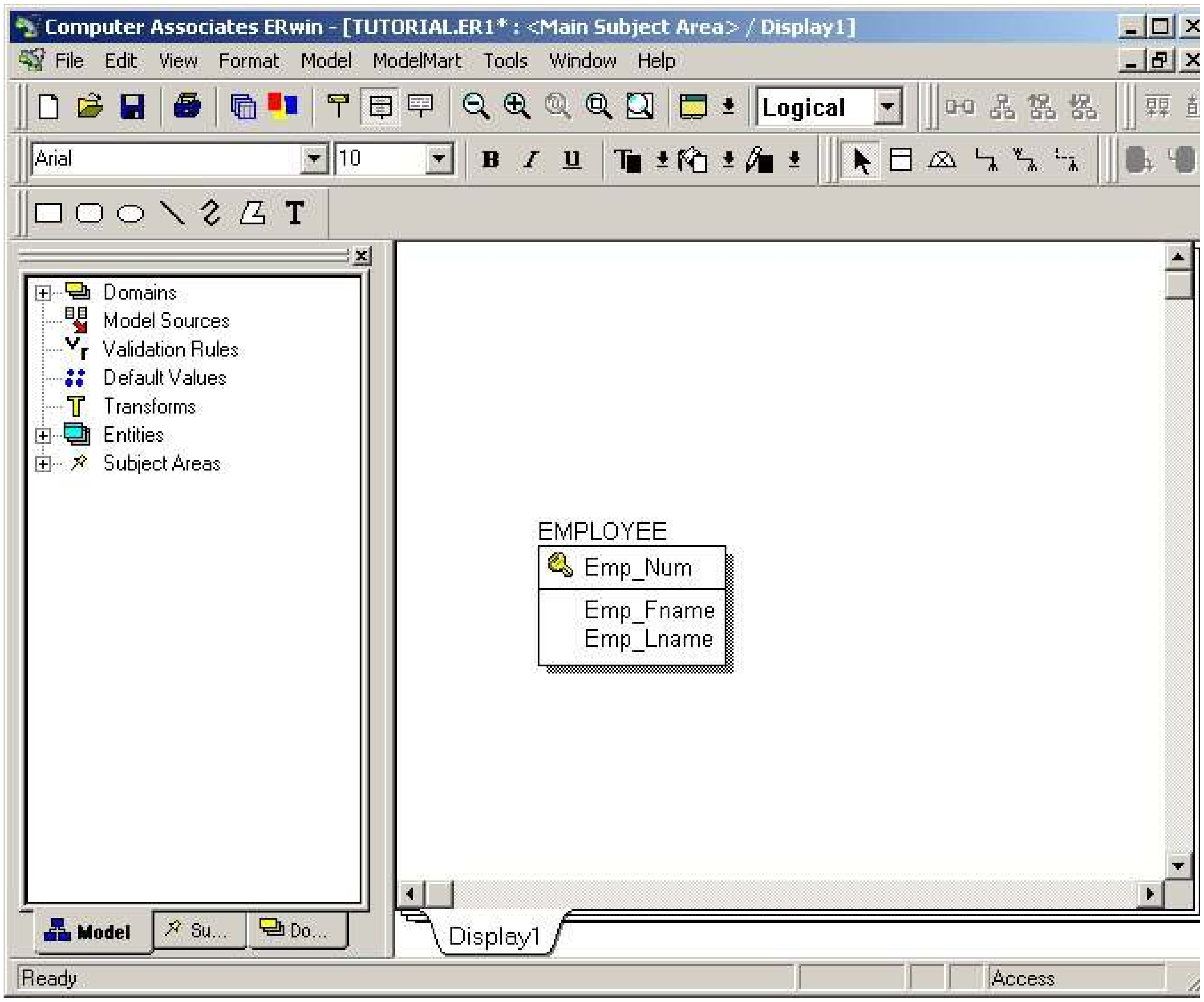
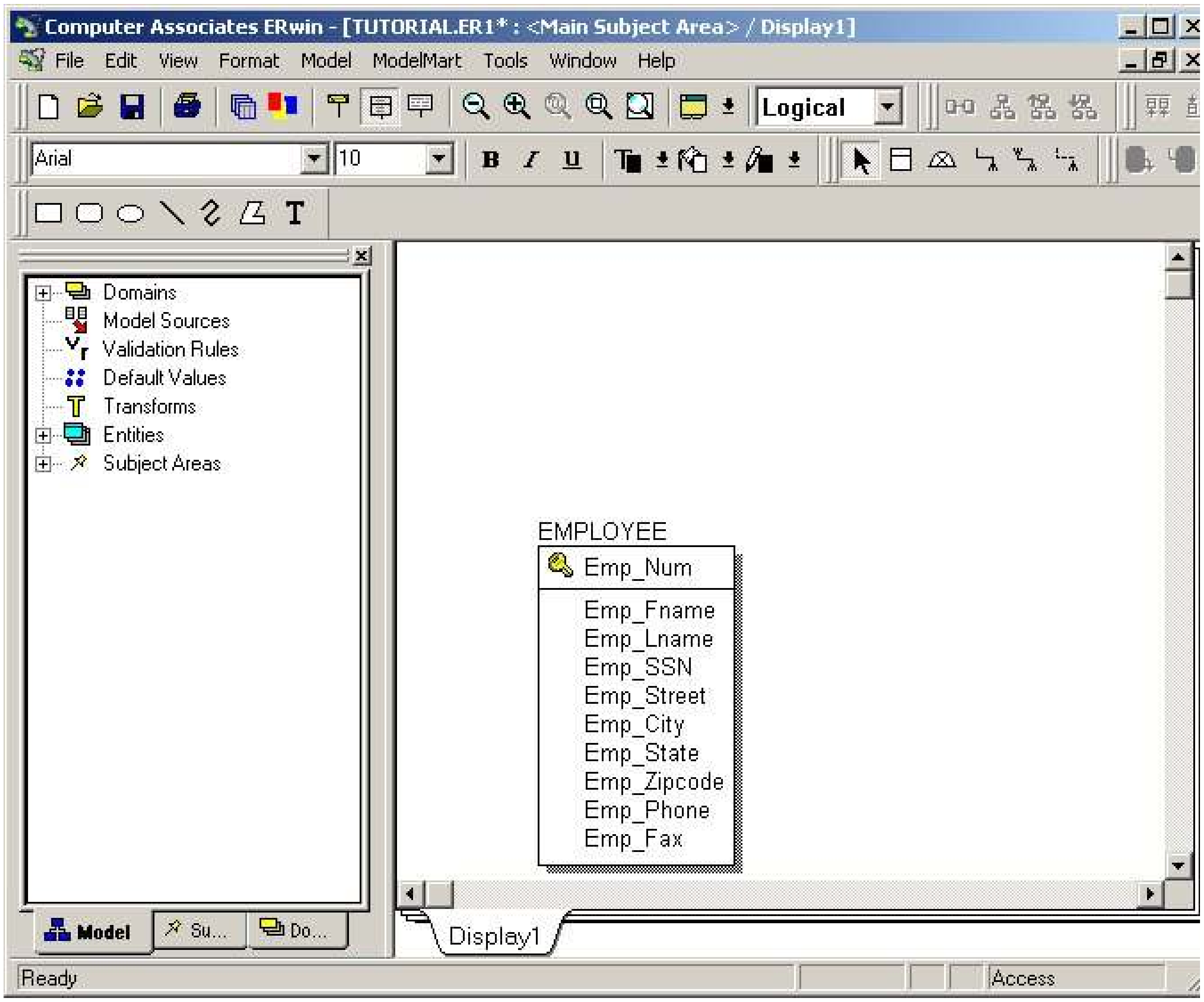


Figure 7: Adding non-primary key attributes

Continue adding the following non-primary key attributes:

Emp_Lname
Emp_SSN
Emp_Street
Emp_City
Emp_State
Emp_Zipcode
Emp_Phone
Emp_Fax

Your diagram should now look like Figure 8.


Computer Associates ERwin - [TUTORIAL.ER1* : <Main Subject Area> / Display1]
File Edit View Format Model ModelMart Tools Window Help
Logical
Arial
10
Domains
Model Sources
Validation Rules
Default Values
Transforms
Entities
Subject Areas
Model
Su...
Do...
Display1
Ready
Access

EMPLOYEE
Emp_Num
Emp_Fname
Emp_Lname
Emp_SSN
Emp_Street
Emp_City
Emp_State
Emp_Zipcode
Emp_Phone
Emp_Fax

Figure 8: The Employee entity

Repeat the entity creation process for the entities and attributes listed in Table 1.

Entity	Attributes	Entity	Attributes
OFFICE	Office_Num (pk) Office_Street Office_City Office_State Office_Zipcode Office_Name Office_Phone Office_Fax	INSPECTION	Insp_Date (pk) Insp_Comments
PROPERTY	Prop_ID (pk) Prop_Street Prop_City Prop_State Prop_Type	MANAGER	Man_Salary Man_Car_Allowance

	Prop_Baths Prop_Rooms Prop_MonthlyRent		
		ASSOCIATE	Assoc_Hourly_Rate

Table 1: Entities and Attributes

Notice that some of the entities have no primary key attributes (Manager, Associate), and one entity has an attribute which is part of the primary key, but by itself does not constitute a primary key (Inspection). As we add relationships to the diagram, the nature of the keys for these entities will become clear. For the time being, be sure that there are no primary key attributes for Manager and Associate, and only one primary key attribute for Inspection, as shown in Figure 9.

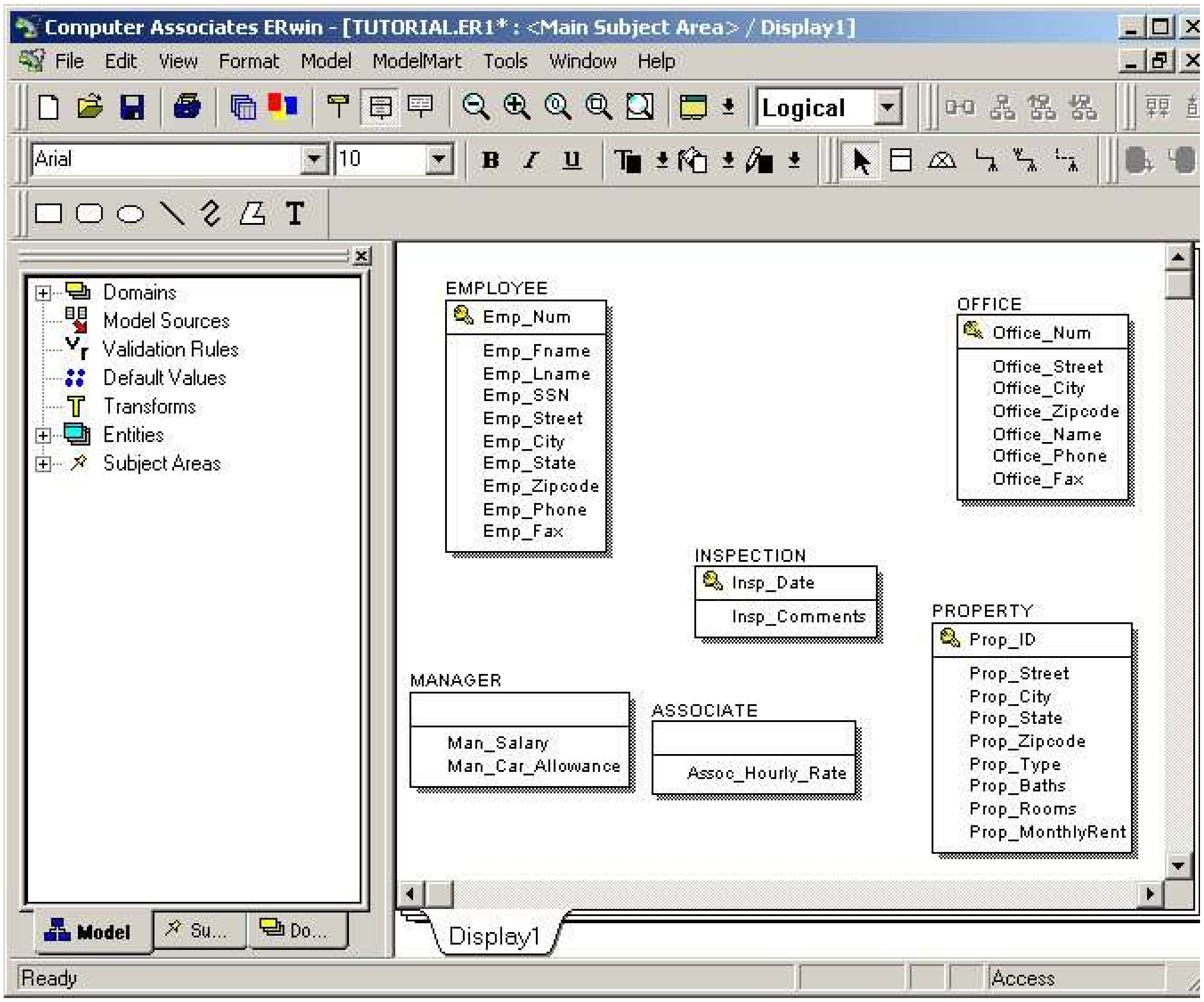


Figure 9: Heartland Properties Entities

Creating Relationships

ER*win* supports the creation of relationships with three basic kinds of connectivity: one-to-one, one-to-many, and many-to-many. Within the one-to-many category, ER*win* allows us to distinguish between *identifying* and *non-identifying* one-to-many relationships.

One-to-many Relationships

We'll begin with two relationships, between Inspection and Employee and between Inspection and Property. Inspection is related to both Property and Employee in one to many relationships. Each Inspection is undertaken at one Property; each Property may undergo many inspections. Each Inspection is carried out by one Employee; each Employee carries out many Inspections. Are these relationships *identifying* relationships or *non-identifying* relationships? The issue hinges on the nature of the primary key of Inspection, the entity on the "many" side of the relationship. The primary key of Inspection in this example is a composite primary key consisting of two attributes: {Insp_Date, Prop_ID} The business rule in effect here is that each property has at most one inspection per day. One might argue whether or not that is a sound assumption, but lets suppose for the sake of illustration that it is. Notice that the primary key of Inspection includes the primary key of Property, the entity with which it has a relationship. For this reason, we say that the relationship between Inspection and Property is an *identifying* one-to-many relationship.

An *identifying* relationship is created by clicking first on the identifying relationship icon ( ).

Notice that it has the crow's foot notation (indicating "many") and a solid line (indicating *identifying*). To create an identifying relationship, click first on this icon, then click on the parent entity (on the one side of the relationship) and then click on the child entity (on the many side of the relationship). In this case, you will click first on the identifying relationship icon, then on Property, then on Inspection. The results are shown in Figure 10.

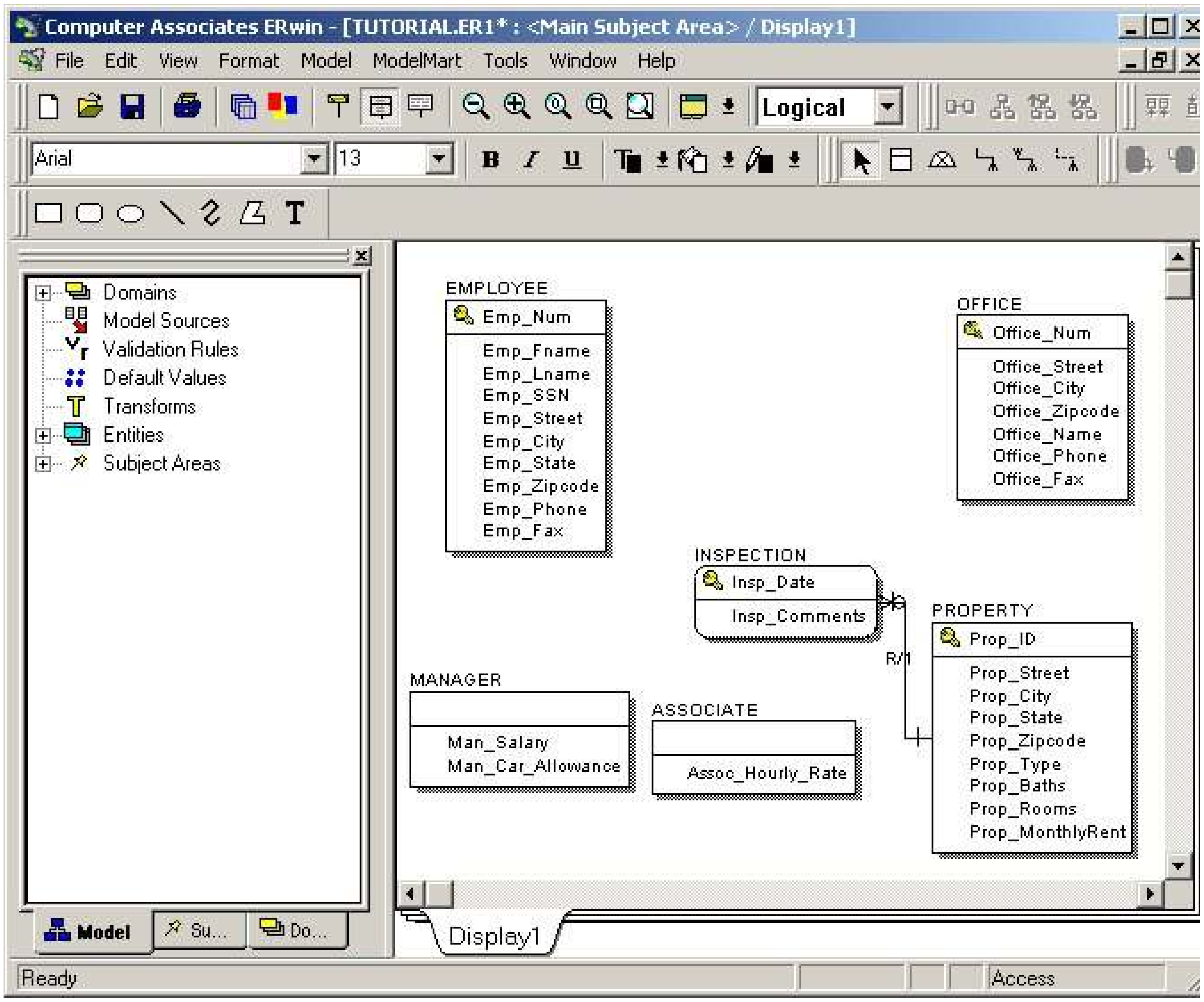
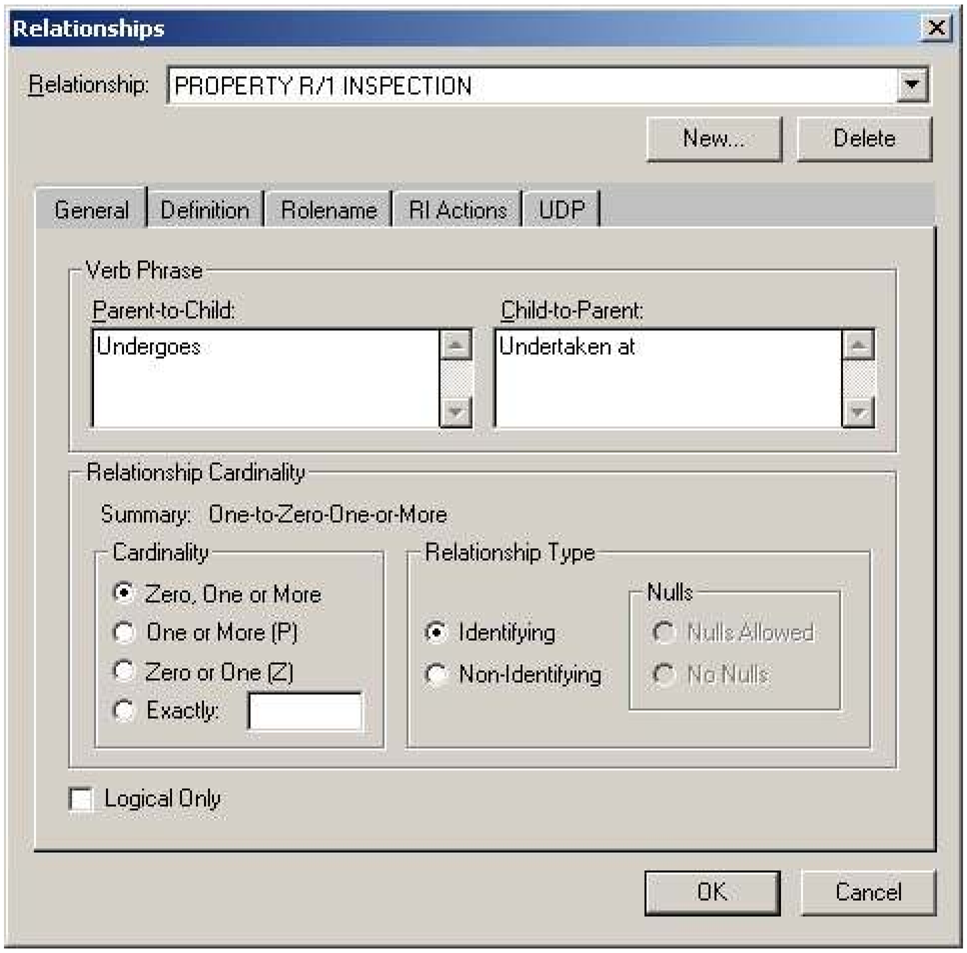


Figure 10: An identifying relationship

Notice that ER*win* has provided a label "R/1" for the relationship, which is not very helpful. Double-click on the relationship itself to bring up a dialog box in which we can further refine the relationship definition. Fill out this dialog box as shown in Figure 11.


Relationships

Relationship: PROPERTY R/1 INSPECTION

Verb Phrase

Parent-to-Child: Undergoes

Child-to-Parent: Undertaken at

Relationship Cardinality

Summary: One-to-Zero-One-or-More

Cardinality: [✓ Zero, One or More] One or More (P) Zero or One (Z) Exactly:

Relationship Type: [✓ Identifying] Non-Identifying

Nulls: Nulls Allowed No Nulls

☐ Logical Only

Figure 11: Relationships definition

In the *Relationship:* text box the relationship is presented in the form *parent entity* R/1 *child entity*. The Verb Phrase portion of this box allows us to define the label to place on the relationship in place of R/1. Since the parent entity is Property and the Child entity is Inspection, the Parent-to-Child verb phrase will read "Property Undergoes Inspection." The Child-to-Parent verb phrase will read "Inspection Undertaken at Property."

In the Relationship Cardinality portion of this window, we can determine how many child entity occurences may be associated with each parent entity occurence. More specifically, "One Property Undergoes *Zero, One or More* Inspections." Notice that at this point we are also able to define participation. If each property had to have undergone at least one inspection in order to be stored in the database, then we could have forced *mandatory* participation by choosing the *One or More* option.

Click OK to complete the relationship definition, as shown in Figure 12.

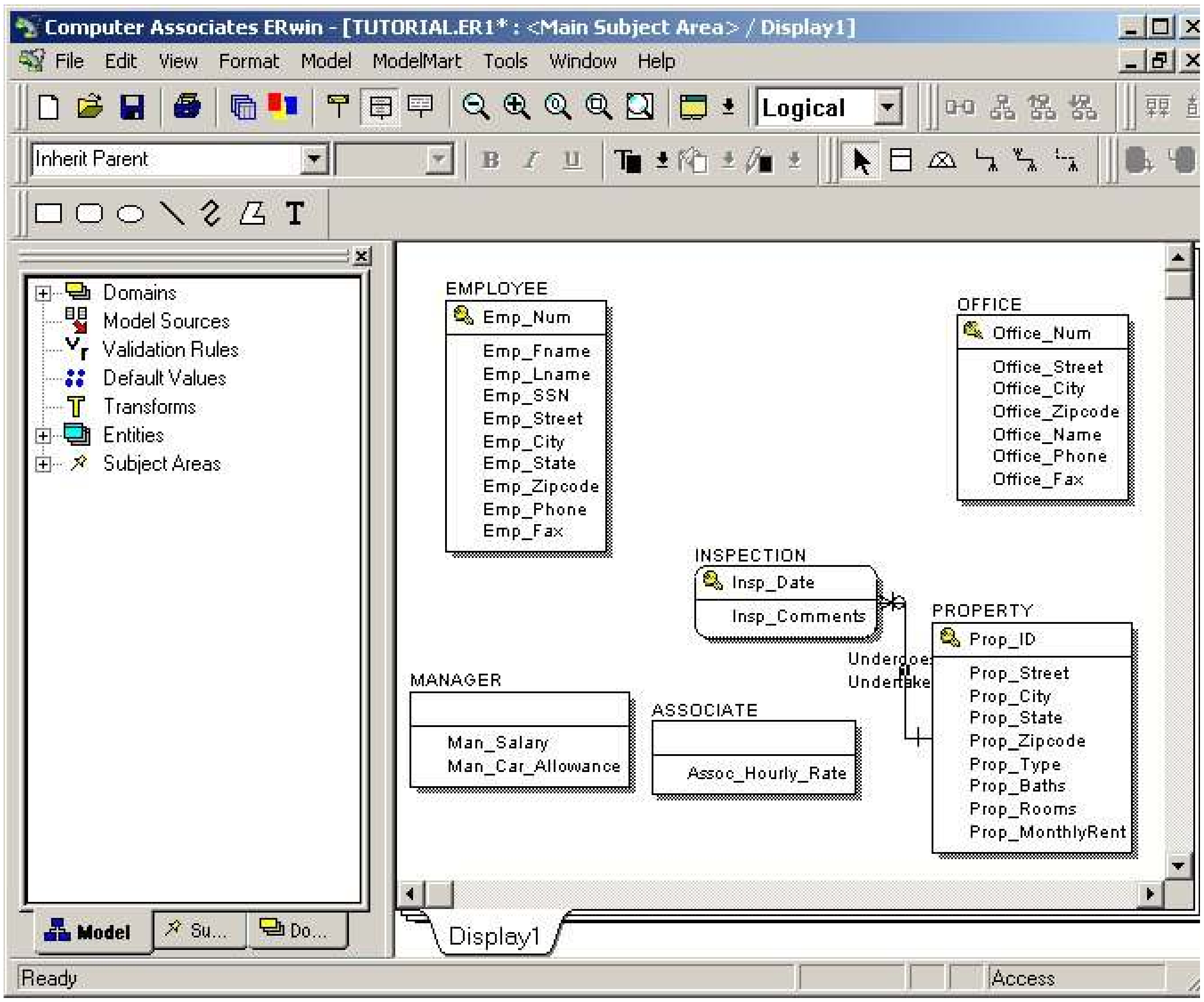
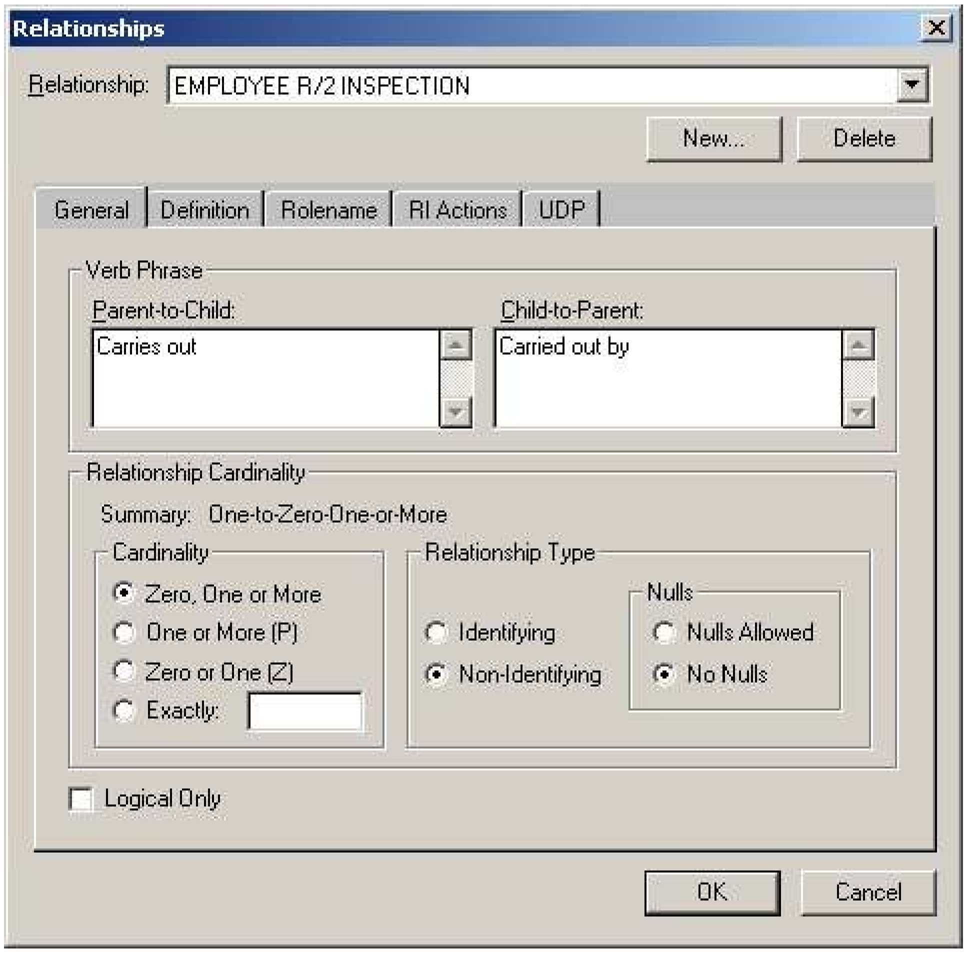


Figure 12: Finished relationship

Notice that when an entity participates as the child in an *identifying relationship* it is modeled with rounded corners. This notation is used for what in other data modeling tools is called a "Weak Entity".

The relationship between Inspection and Employee is somewhat different. It is a one-to-many relationship, but it is *non-identifying*, because the primary key of Employee is *not* part of the primary key of Inspection. For such a relationship, we must click on the non-identifying relationship icon ( ), which uses a dashed line instead of a solid line.

As before, we click on the icon, then the parent entity, then the child entity. Click on the non-identifying relationship icon, then the Employee entity, then the Inspection entity. Double-click on the relationship to bring up the relationship definition window and fill it in as shown in Figure 13.


Relationships

Relationship: EMPLOYEE R/2 INSPECTION

Verb Phrase

Parent-to-Child: Carries out

Child-to-Parent: Carried out by

Relationship Cardinality

Summary: One-to-Zero-One-or-More

Cardinality: [✓ Zero, One or More] One or More (P) Zero or One (Z) Exactly:

Relationship Type: Identifying [✓ Non-Identifying]

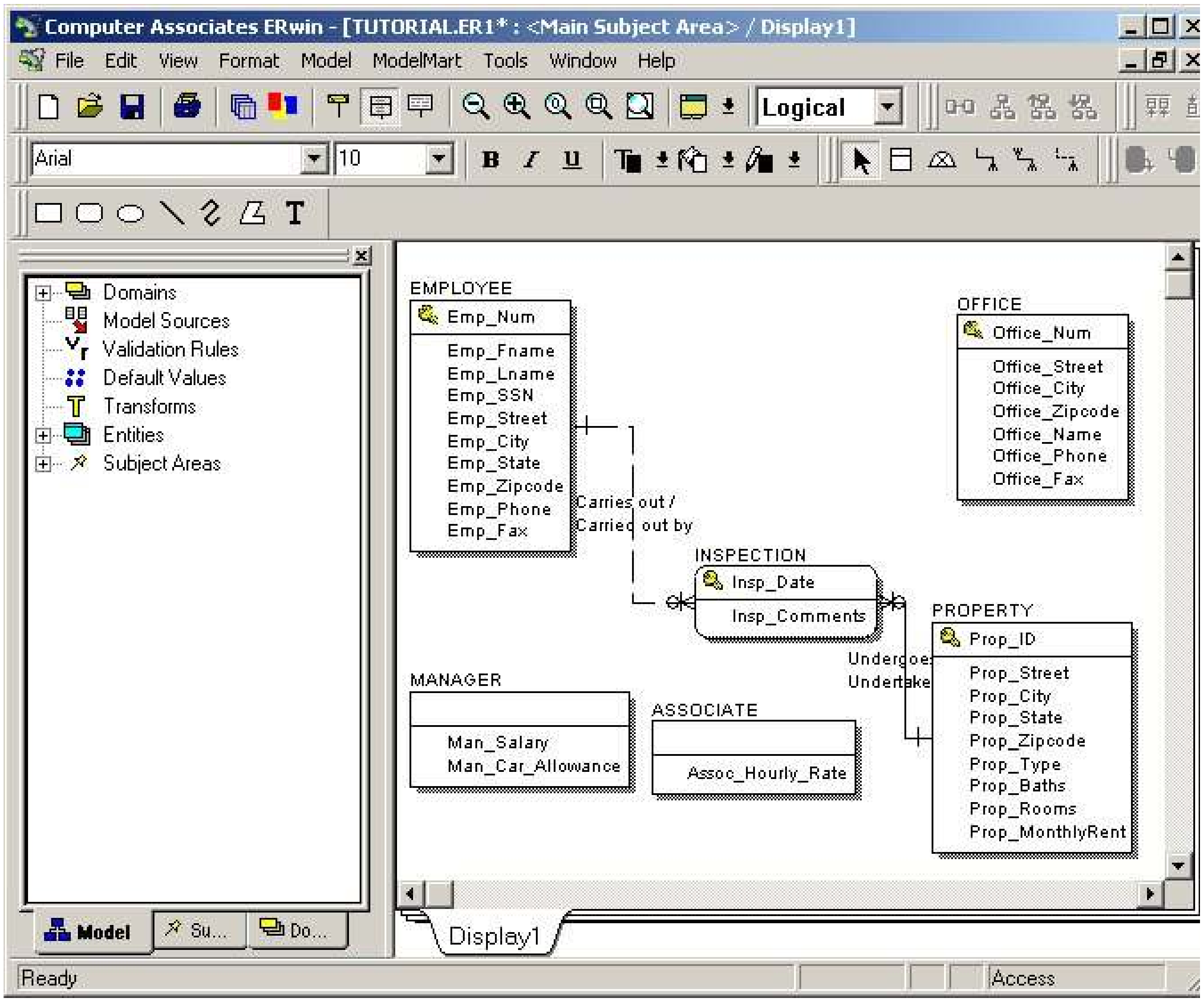
Nulls: Nulls Allowed [✓ No Nulls]

Logical Only: ☐

Figure 13: Non-identifying relationship definition.

As before, we provide more meaningful verb phrases for the relationship. As before, since each employee may be associated with zero, one or more inspections, we choose the corresponding cardinality. Unlike before, when we chose the identifying relationship type, we have now chosen the non-identifying relationship type. We have the option of indicating whether nulls are permitted or not. This decision regards the participation of the parent entity in a relationship with the child entity. In other words, does each child entity occurence (Inspection) *have to be* associated with a parent entity occurence (Employee)? In this case, the answer yes. Each inspection must be carried out by an employee, or it is not considered a proper inspection. The participation of Employee in the relationship is *mandatory*.

Click OK to complete the relationship definition, as shown in Figure 14.


Computer Associates ERwin - [TUTORIAL.ER1* : <Main Subject Area> / Display1]
File Edit View Format Model ModelMart Tools Window Help
Logical
Arial
10
Domains
Model Sources
Validation Rules
Default Values
Transforms
Entities
Subject Areas
Model
Su...
Do...
Display1
Ready
Access

EMPLOYEE
Emp_Num
Emp_Fname
Emp_Lname
Emp_SSN
Emp_Street
Emp_City
Emp_State
Emp_Zipcode
Emp_Phone
Emp_Fax

OFFICE
Office_Num
Office_Street
Office_City
Office_Zipcode
Office_Name
Office_Phone
Office_Fax

Carries out /
Carried out by

INSPECTION
Insp_Date
Insp_Comments

PROPERTY
Prop_ID
Prop_Street
Prop_City
Prop_State
Prop_Zipcode
Prop_Type
Prop_Baths
Prop_Rooms
Prop_MonthlyRent

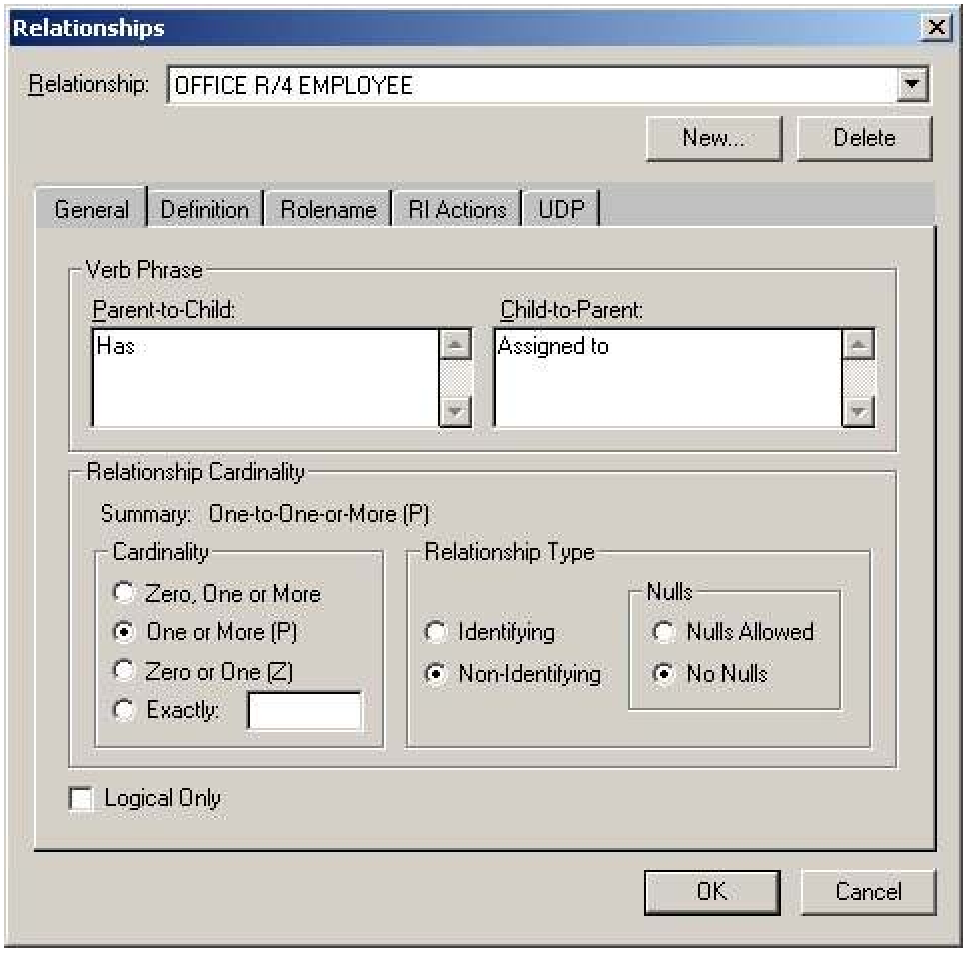
Undergoes
Undertake

MANAGER
Man_Salary
Man_Car_Allowance

ASSOCIATE
Assoc_Hourly_Rate

Figure 14: Non-identifying relationship

Employee and Office also participation in a relationship with each other. Each Employee is assigned to one and only one Office, and each Office has one or more Employees. In this one-to-many relationship, Office is the parent and Employee is the child. Since the primary key of the child (Employee) does not include the primary key of the parent (Office), this is a non-identifying relationship. Click on the non-identifying relationship icon, Office, and Employee to create this relationship. Double-click on the relationship and fill in the relationship definition window as shown in Figure 15.


Relationships

Relationship: OFFICE R/4 EMPLOYEE

Verb Phrase
Parent-to-Child: Has
Child-to-Parent: Assigned to

Relationship Cardinality
Summary: One-to-One-or-More (P)

Cardinality: ( ) Zero, One or More [✓ One or More (P)] ( ) Zero or One (Z) ( ) Exactly:

Relationship Type: ( ) Identifying [✓ Non-Identifying]

Nulls: ( ) Nulls Allowed [✓ No Nulls]

☐ Logical Only

Figure 15: A non-identifying relationship with mandatory participation

In Figure 15, see an example of a relationship in which participation is mandatory for both entities. The Cardinality indicates that each Office is associated with One or More (but not Zero) Employees. The Relationship Type indicates that this is a non-identifying relationship, and that Nulls are not permitted. In other words, each child (Employee) *must* be associated with a single parent (Office).

Complete the relationships by creating a relationship between Office and Property in which each Office manages zero, one, or more properties, and each property is managed by one and only one Office. Your diagram should now appear as shown in Figure 16.

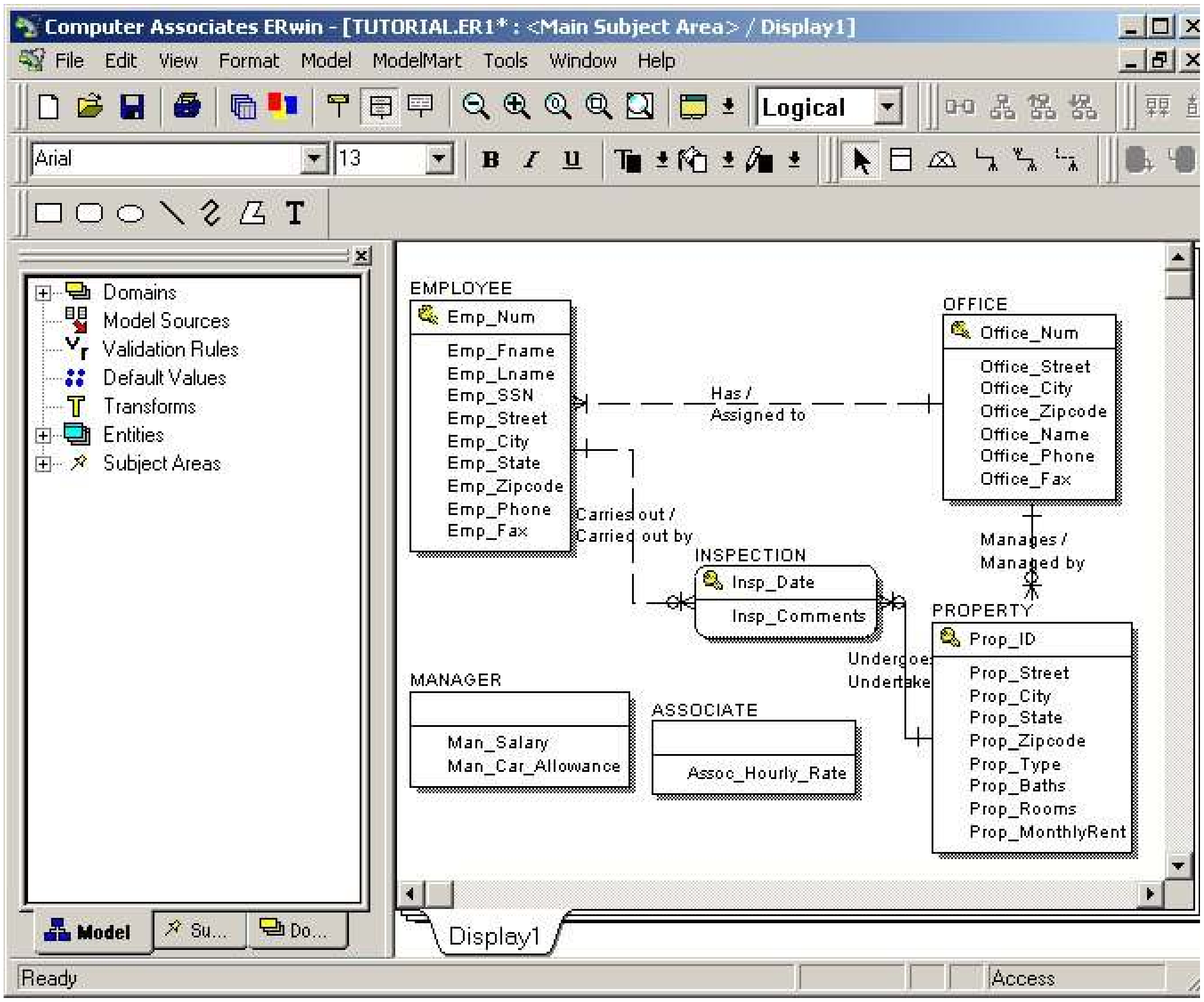


Figure 16: Relationships

Subtypes and Supertypes

There are two kinds of employees we wish to distinguish between: Managers and Associates. Each of these two kinds of employees has all of the properties of Employee, but have in addition a small number of specific attributes. Only managers have a salary and a car allowance. Only associates has an hourly rate. We can specify that Manager and Associate are subtypes of Employee. To do this, we will use the subtype icon ( ). To indicate that Manager is a subtype of Employee, click first on the subtype icon, then on the supertype (Employee), then on the subtype (Manager). The result appears in Figure 17.

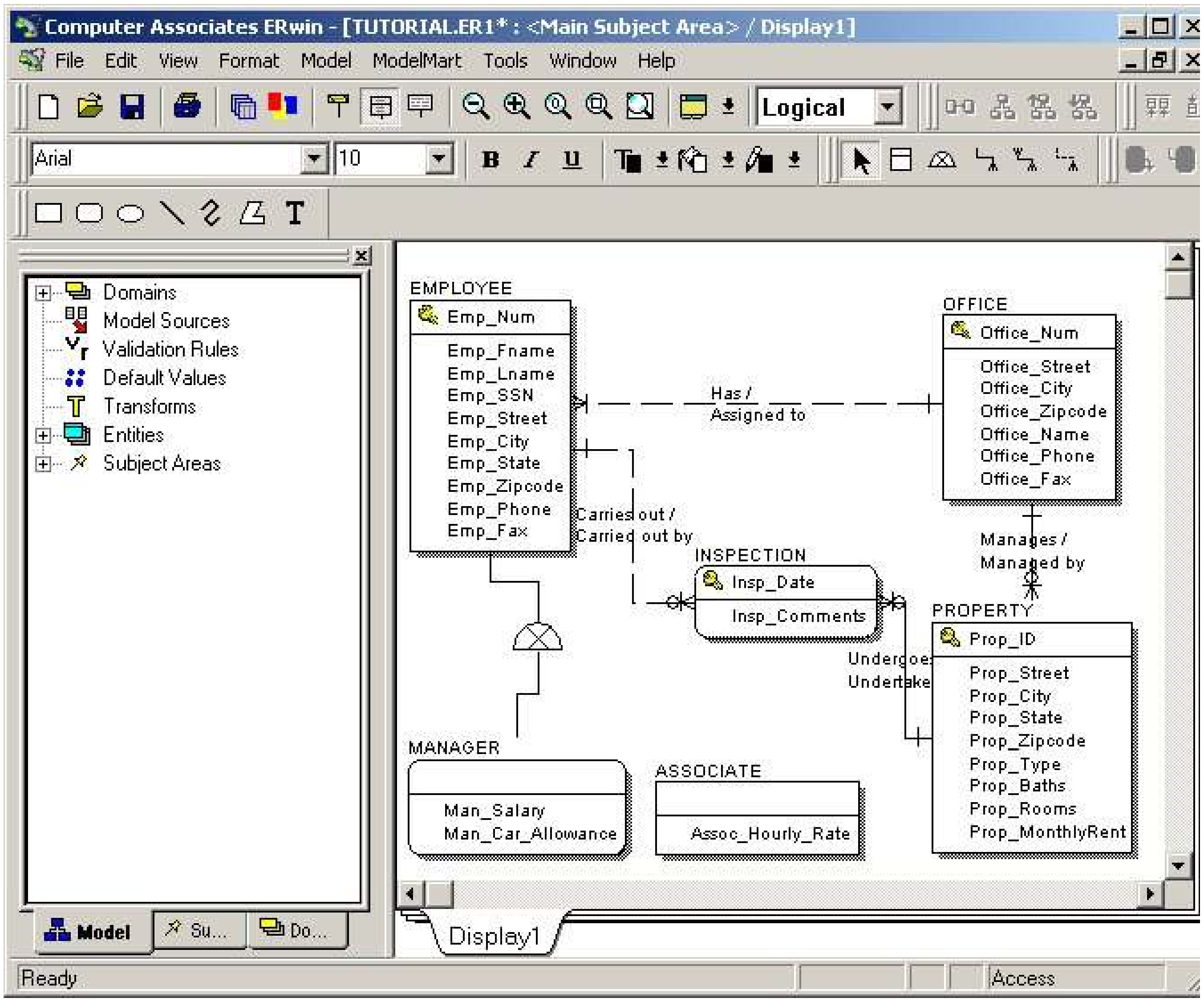


Figure 17: Subtype relationship

To include Associate as a subtype, click again on the subtype icon. *Now, instead of clicking on the supertype again, click on the subtype icon in the diagram between Employee and Manager. Then, click on Associate*. The result appears in Figure 18.

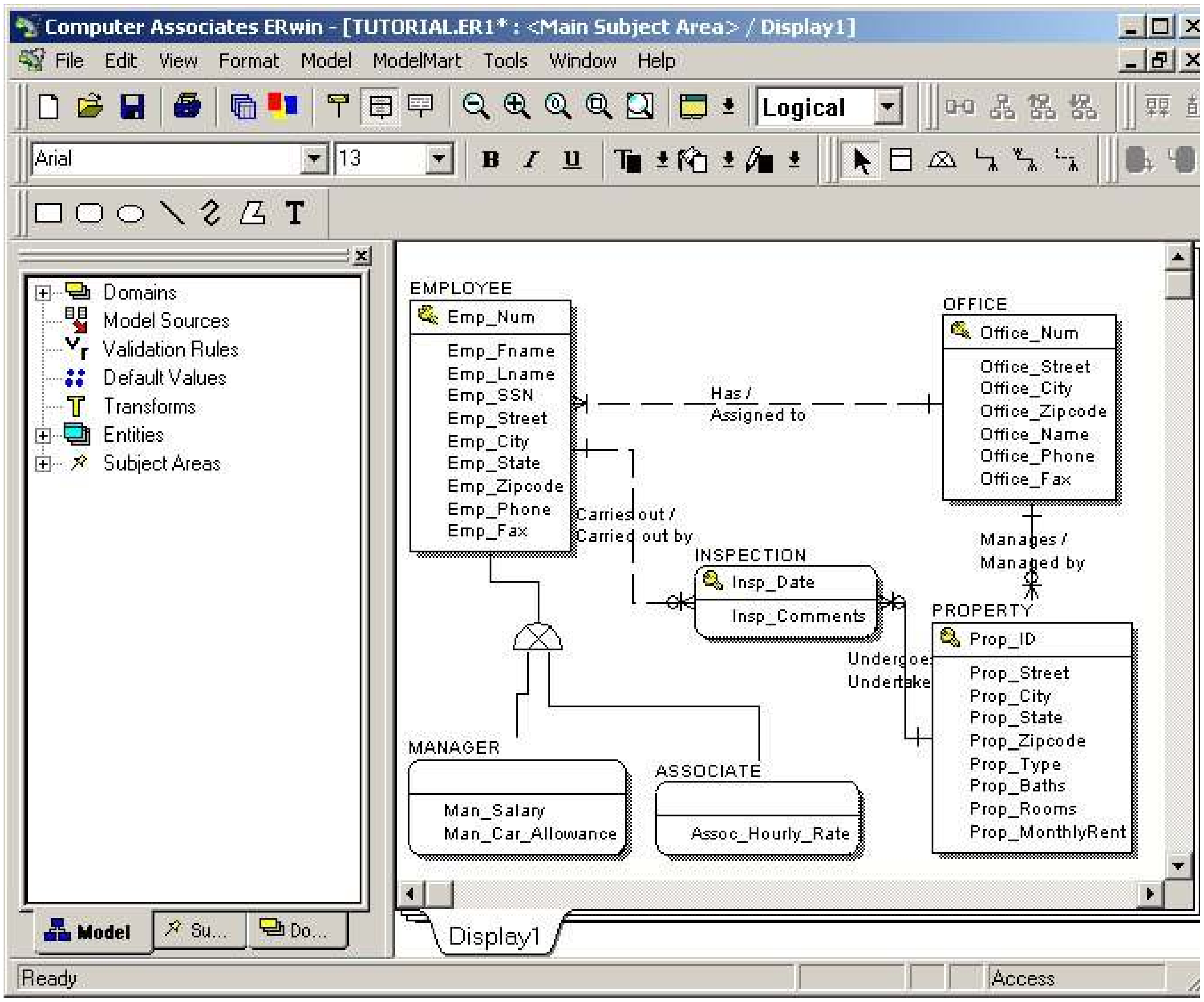


Figure 18: The Final ER Diagram

Finally, save your work. You may now either exit ER*win*, or proceed to the next section, entitled Creating a Relational Schema.