National University of Computer and Emerging Sciences

Lab Manual

“Introduction to Data Modeling Tutorial Using Erwin”

Database Systems

Spring 2023

Department of Computer Science
FAST-NU, Lahore, Pakistan

Table of Contents

1 Objective

The object of this manual is to give an overview and hands on practice to Data modeling using ERWIN

2 Task Distribution

Total Time	50 Minutes
Demo and practice In Lab Question	20 Minutes 30 Minutes

3

Introduction

ERwin is a popular data modeling tool. 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. This brief tutorial steps you through the process of creating a data model using Er*win*. Creation of a basic data model (Conceptual data model)

4 Creation of a basic data model

The Entities involved in this model include: Employee, Department.
From the File Menu choose to create a new model: File->New

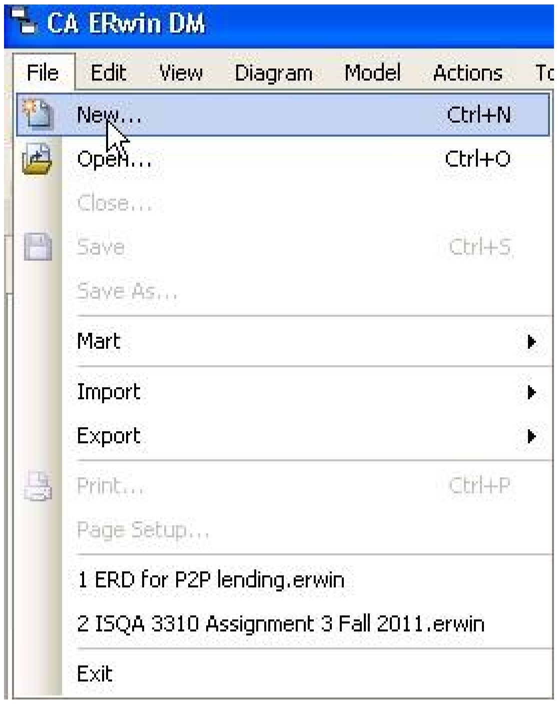
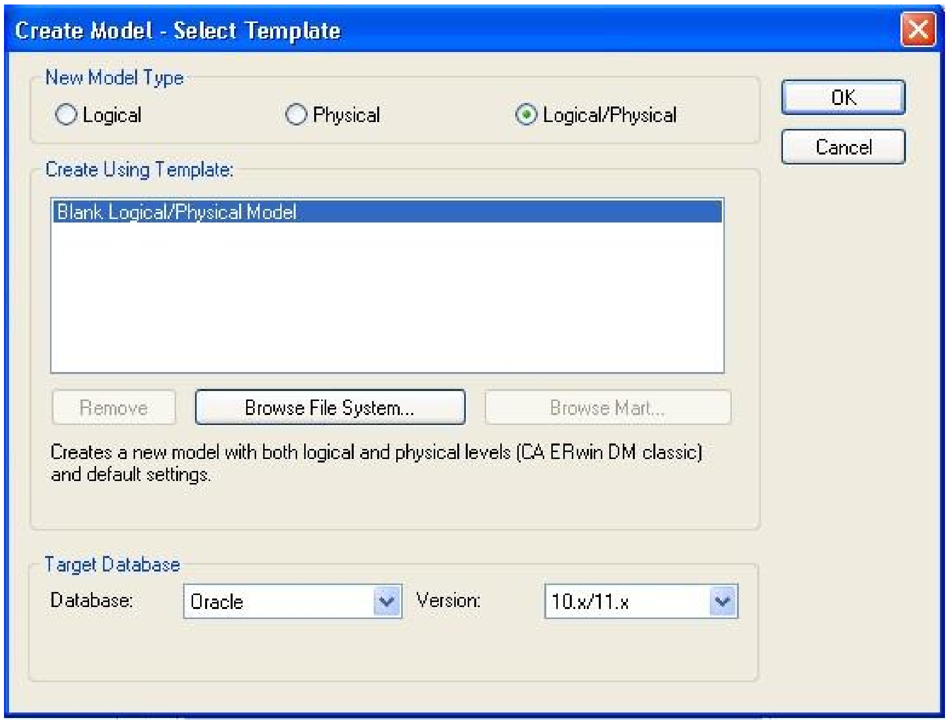


Figure 1: Create a new model

The next dialog box, shown in Figure 2, will ask you to choose the template to be used to create the new model. 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 (CA ERwin DM classic) and default settings.

Target Database — Database: Oracle Version: 10.x/11.x

Figure 2: Selecting a model type

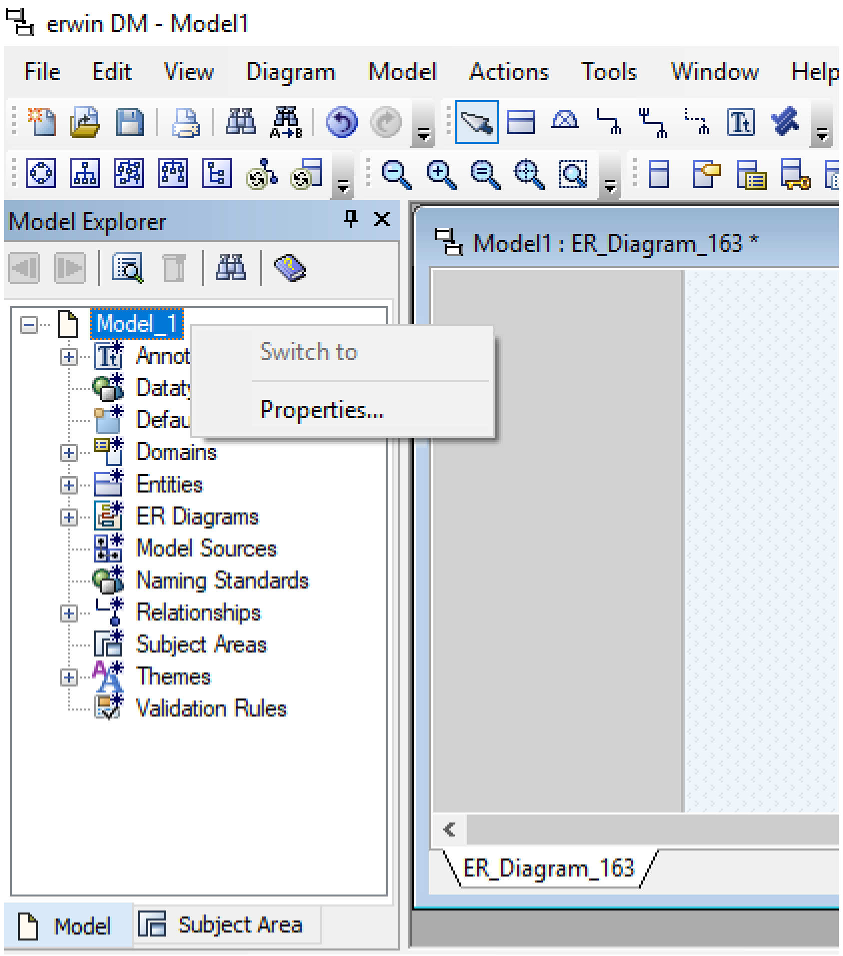
ERwin will now display the main window from which most of your ER diagram development will be done, as shown in Figure 3.

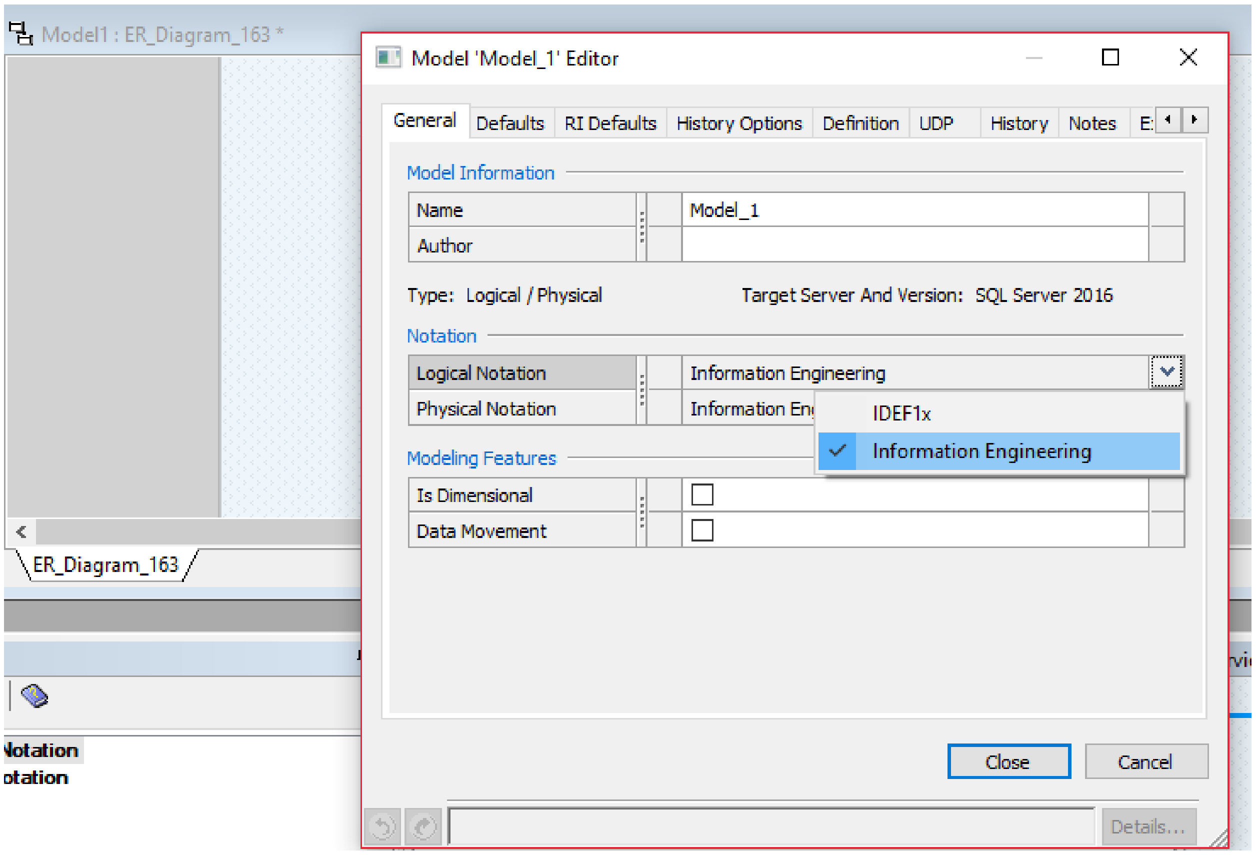
How to Change to IE (Information Engineering) Notation

Using ERwin DM, how do I change my relationship's so they have Information Engineering (IE) notation?

Solution:

Please go to the **Model | model properties**, **General** tab. You can change the notation by clicking the drop-down arrow at the right of Logical Notation and Physical Notation. Then select Information Engineering.


erwin DM - Model1
File
Edit
View
Diagram
Model
Actions
Tools
Window
Help
Model Explorer
Model_1
Annot
Dataty
Defau
Domains
Entities
ER Diagrams
Model Sources
Naming Standards
Relationships
Subject Areas
Themes
Validation Rules
Switch to
Properties...
Model1 : ER_Diagram_163 *
ER_Diagram_163
Model
Subject Area



Model Information

Name	Model_1
Author	

Type: Logical / Physical   Target Server And Version: SQL Server 2016

Notation

Logical Notation	Information Engineering
Physical Notation	Information En

IDEF1x
✓ Information Engineering

Modeling Features

Is Dimensional	☐
Data Movement	☐

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.

5 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.

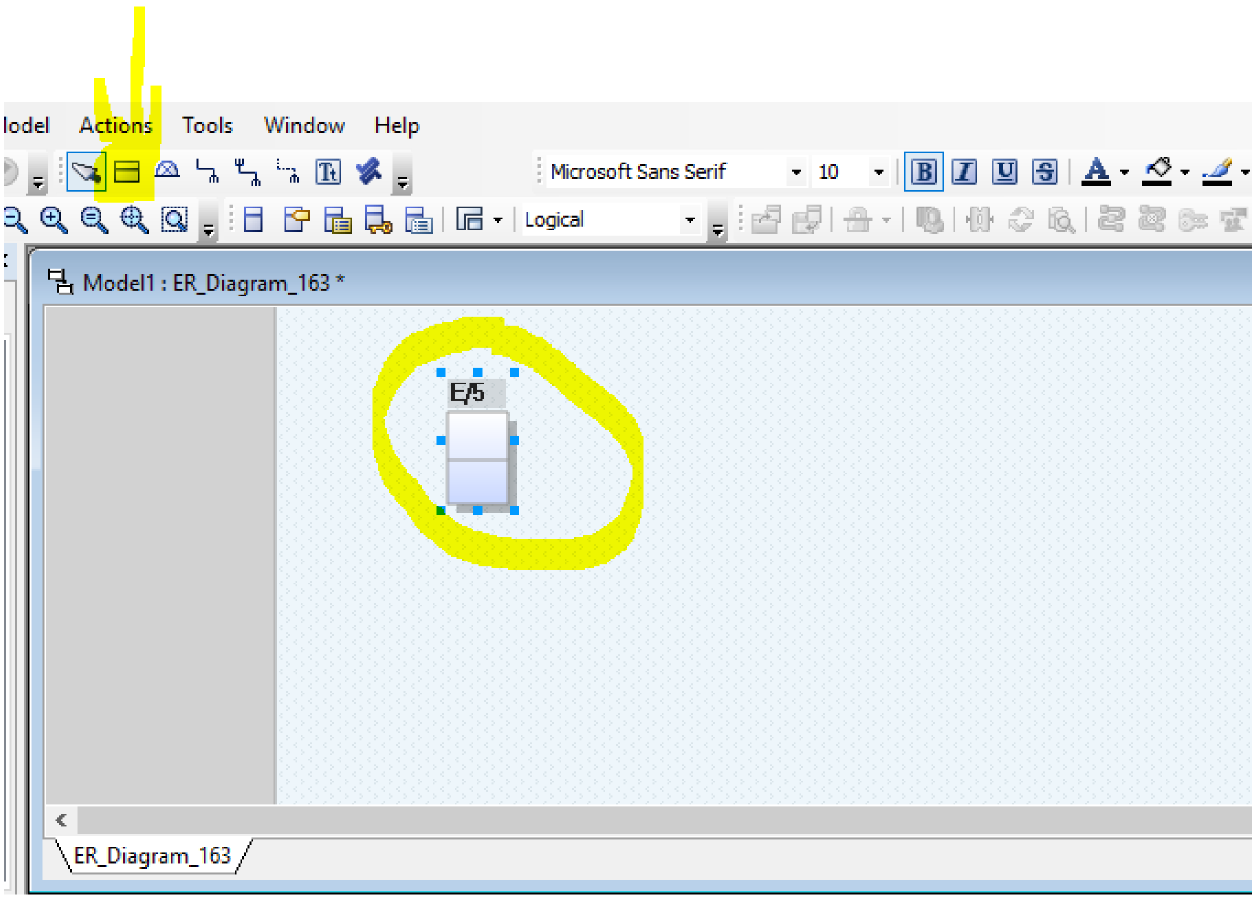


Figure 4: A new entity

Notice that the default name for the entity is E/x, where x is some number (1 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.

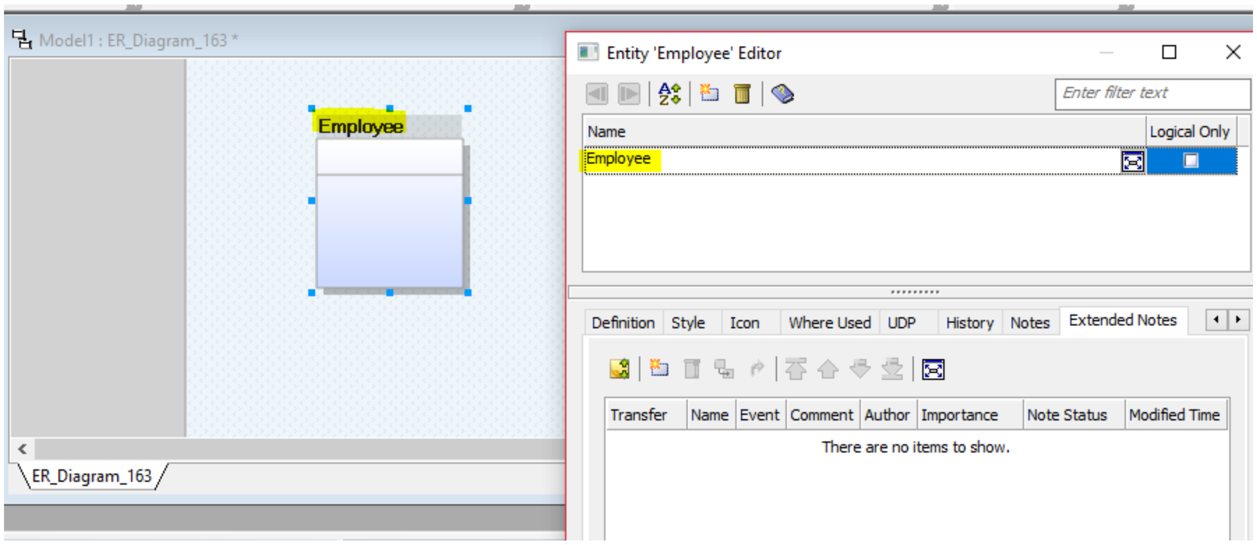
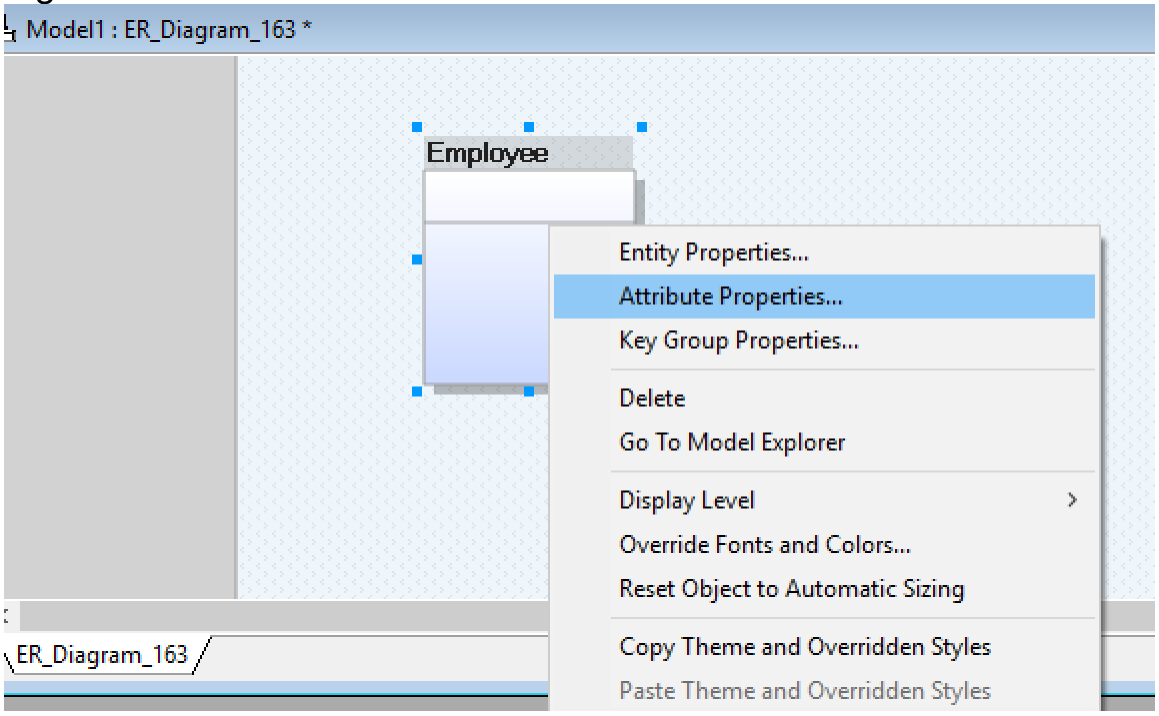


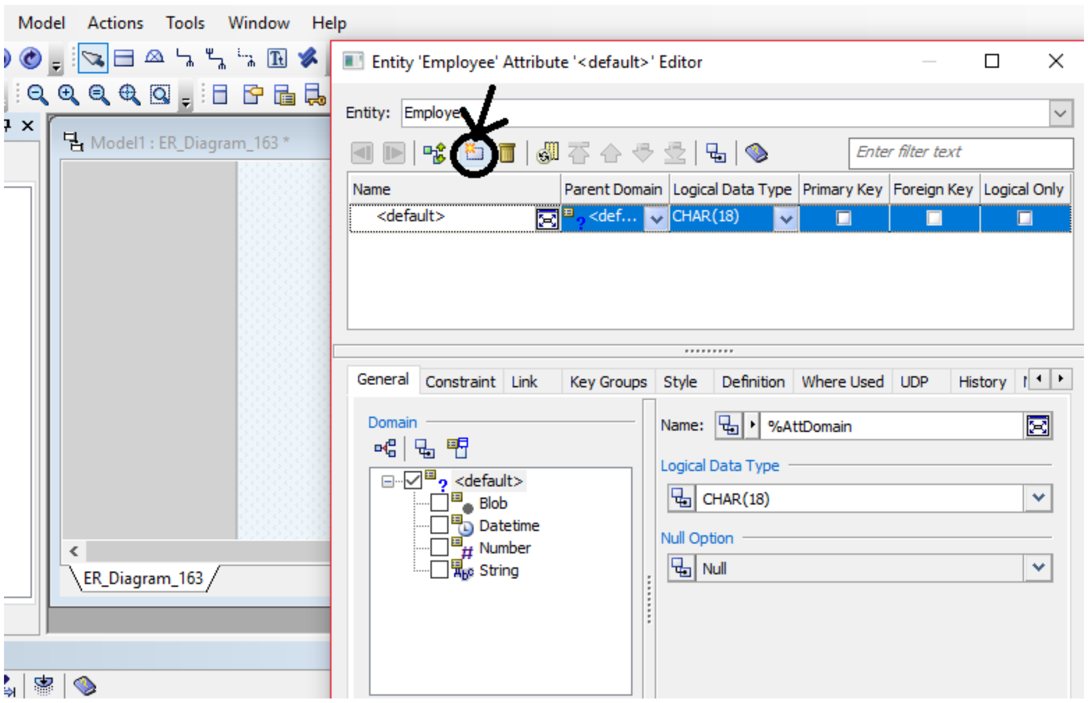
Figure 5: Changing the name of the entity

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

6 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, **id**, as shown in Figure 6.


Model1 : ER_Diagram_163 *
Employee
Entity Properties...
Attribute Properties...
Key Group Properties...
Delete
Go To Model Explorer
Display Level
Override Fonts and Colors...
Reset Object to Automatic Sizing
Copy Theme and Overridden Styles
Paste Theme and Overridden Styles
ER_Diagram_163



Entity 'Employee' Attribute '<default>' Editor

Entity: Employee

Enter filter text

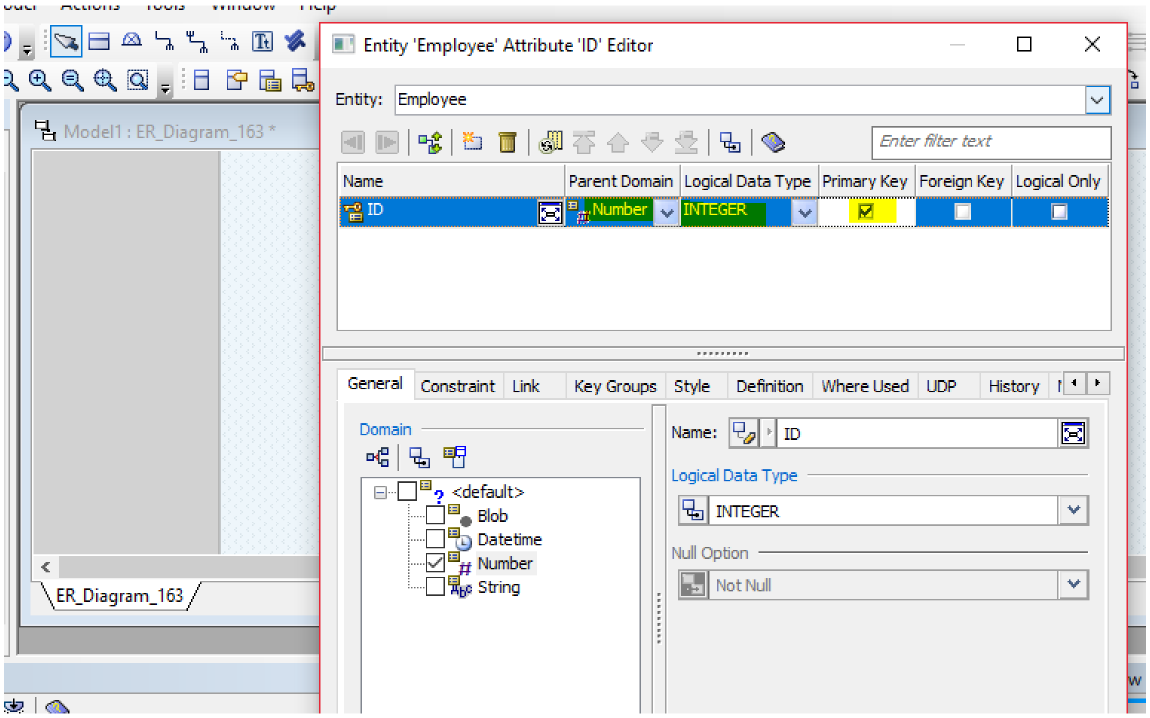
Name	Parent Domain	Logical Data Type	Primary Key	Foreign Key	Logical Only
<default>	<def…	CHAR(18)	☐	☐	☐

Domain: <default>

Name: %AttDomain

Logical Data Type: CHAR(18)

Null Option: Null



Entity 'Employee' Attribute 'ID' Editor

Entity: Employee

Enter filter text

Name	Parent Domain	Logical Data Type	Primary Key	Foreign Key	Logical Only
ID	Number	INTEGER	☑	☐	☐

Domain: ☑ Number

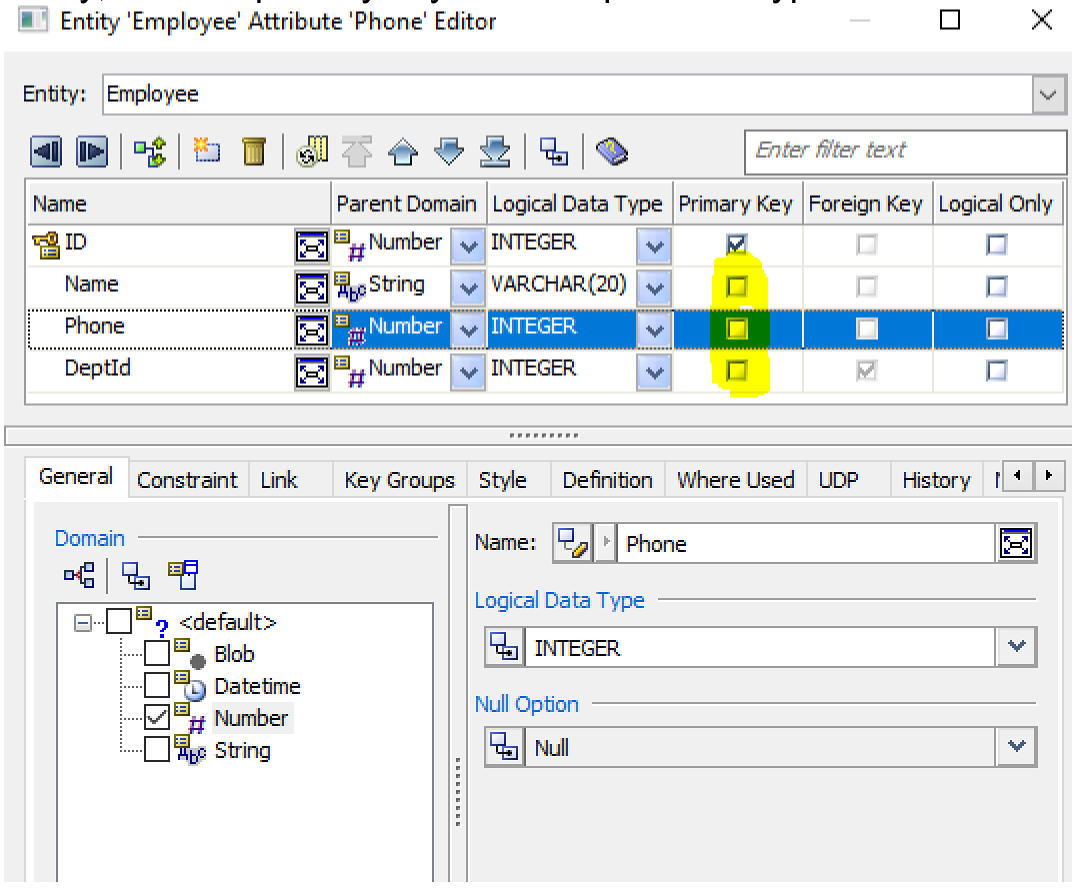
Name: ID

Logical Data Type: INTEGER

Null Option: Not Null

Figure 6: Adding a primary key attribute

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 Name, as shown in Figure 7. When you have typed 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. Type another attribute Phone.


Entity 'Employee' Attribute 'Phone' Editor

Entity: Employee

Enter filter text

Name	Parent Domain	Logical Data Type	Primary Key	Foreign Key	Logical Only
ID	Number	INTEGER	✓	☐	☐
Name	String	VARCHAR(20)	☐	☐	☐
Phone	Number	INTEGER	☐	☐	☐
DeptId	Number	INTEGER	☐	✓	☐

General | Constraint | Link | Key Groups | Style | Definition | Where Used | UDP | History

Domain
- <default>
  - Blob
  - Datetime
  - ✓ Number
  - String

Name: Phone

Logical Data Type: INTEGER

Null Option: Null

Figure 7: Adding non-primary key attributes

Make another entity, DEPARTMET with primary key deptId and attributes name and city.

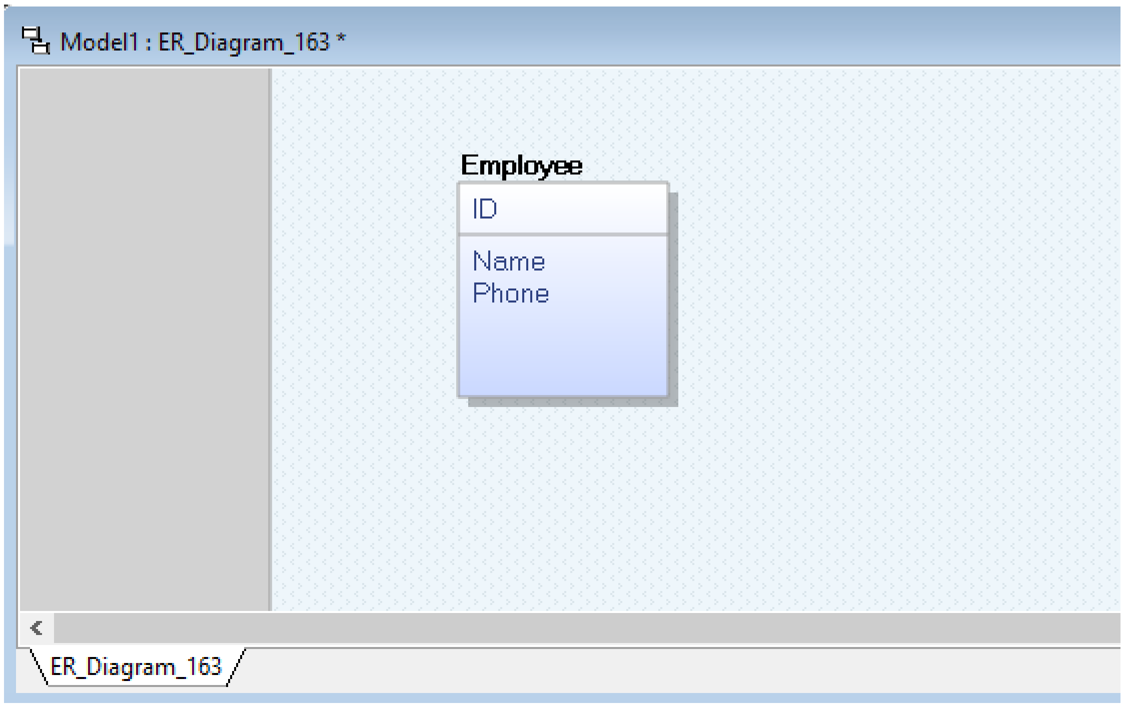


Figure 8: The Employee entity

7 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.

Q: What's the difference between identifying and non-identifying relationships?

- An **identifying relationship** is when the existence of a row in a child table depends on a row in a parent table. This may be confusing because it's common practice these days to create a pseudokey for a child table, but *not* make the foreign key to the parent part of the child's primary key. Formally, the "right" way to do this is to make the foreign key part of the child's primary key. But the logical relationship is that the child cannot exist without the parent.

Example: A Person has one or more phone numbers. If they had just one phone number, we could simply store it in a column of Person. Since we want to support multiple phone numbers, we make a second table PhoneNumbers, whose primary key includes the person_id referencing the Person table.

We may think of the phone number(s) as belonging to a person, even though they are modeled as attributes of a separate table. This is a strong clue that this is an identifying relationship (even if we don't literally include person_id in the primary key of PhoneNumbers).

- A **non-identifying relationship** is when the primary key attributes of the parent *must not* become primary key attributes of the child. A good example of this is a lookup table, such as a foreign key onPerson.state referencing the primary key of States.state. Person is a child table with respect to States. But a row in Person is not identified by its state attribute. I.e. state is not part of the primary key of Person.

A non-identifying relationship can be **optional** or **mandatory**, which means the foreign key column allows NULL or disallows NULL, respectively.

7.1 One-to-many Relationships

An *non-identifying* relationship is created by clicking first on the non-identifying relationship icon ( ). To create an *non-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 *non-identifying* relationship icon, then on DEPARTMENT, then on EMPLOYEE. The results are shown in Figure 10.

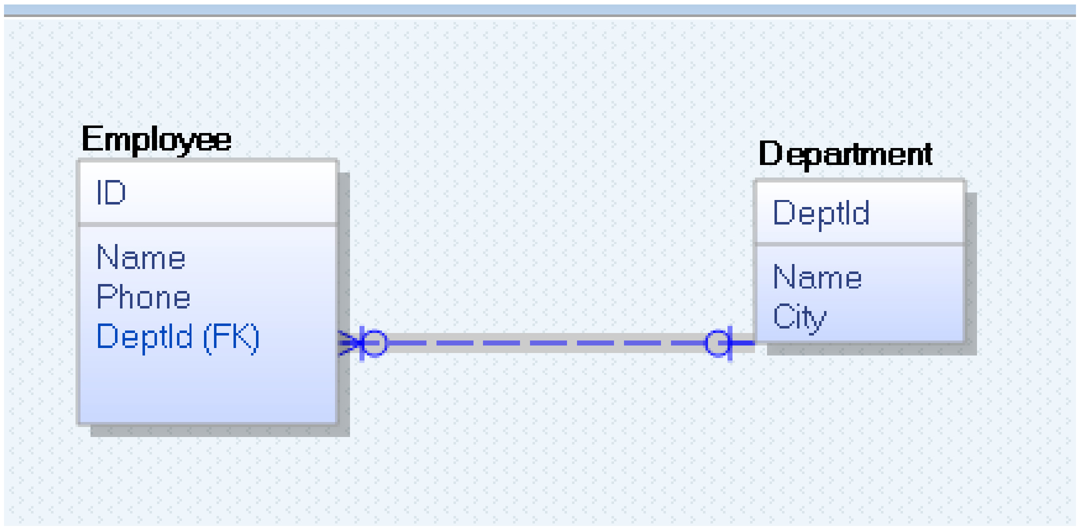
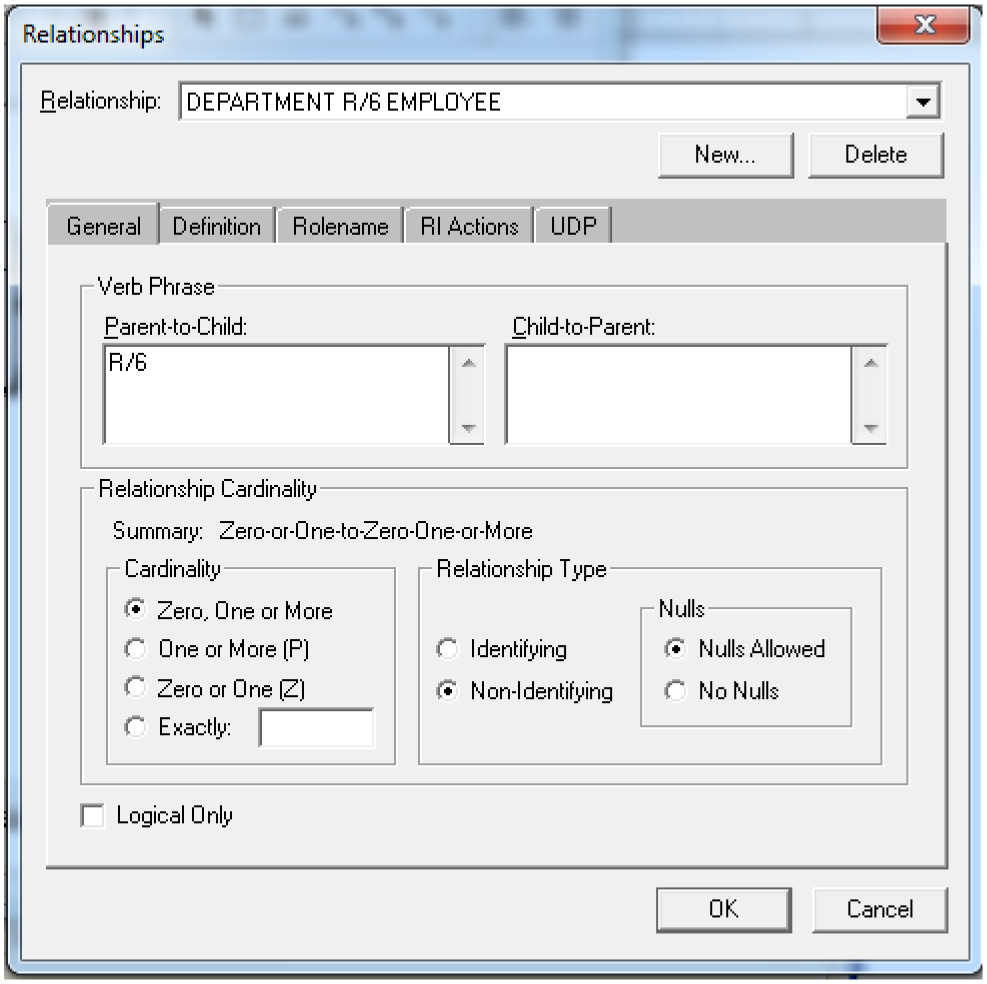


Figure 10: A non-identifying relationship

Double-click on the relationship itself to bring up a dialog box in which we can further refine the relationship definition. As shown in Figure 11.



Relationship: DEPARTMENT R/6 EMPLOYEE

Verb Phrase

Parent-to-Child: R/6

Child-to-Parent:

Relationship Cardinality

Summary: Zero-or-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

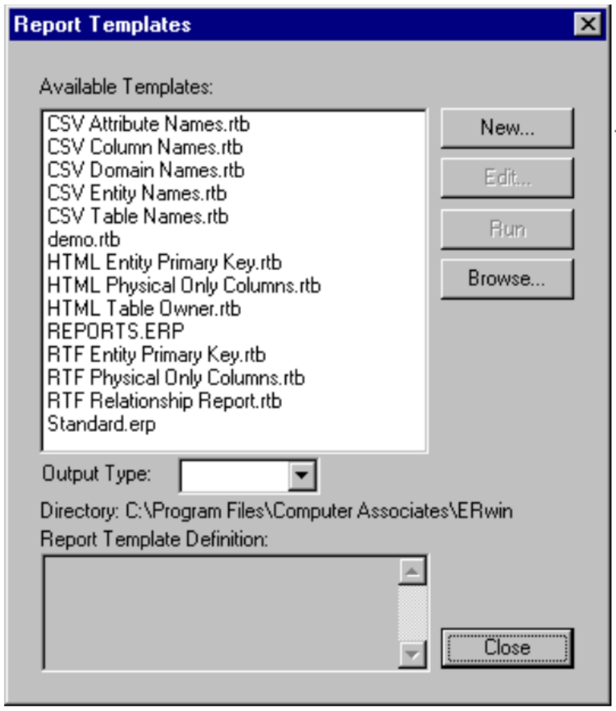
Figure 11: Relationships definition

In the Relationship Cardinality portion of this window, we can determine how many child entity occurrences may be associated with each parent entity occurrence.

You can further inquire about the functionality of Erwin By going though the guide guide provided in Lab 13 folder.

8 Generate a Data Dictionary Using Erwin

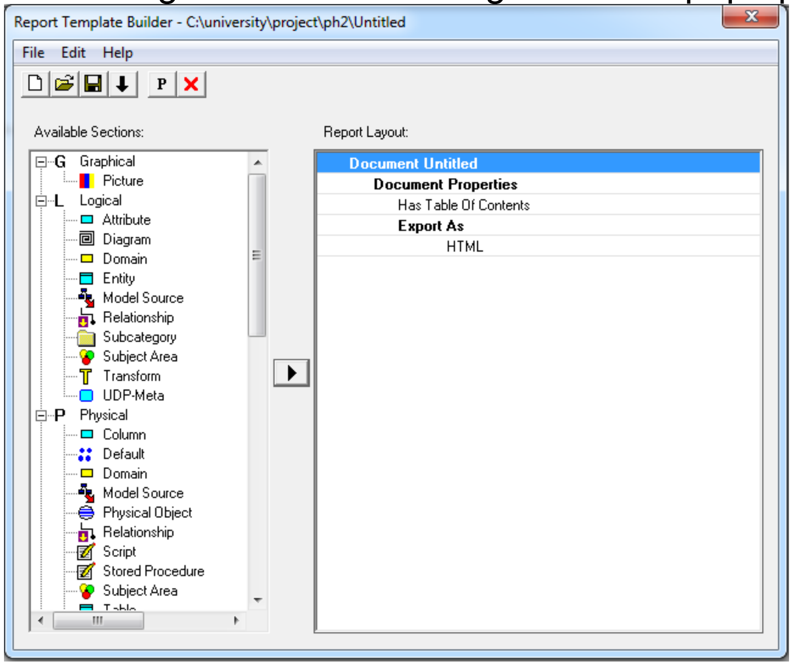
Switch to logical view. Go to the Tools menu, choose Report Builder. Until the first template is saved in the Reports folder, you may see a message indicating that the folder is empty. Or you may see a message indicating that you need to click the Browse button on the next dialog and select a folder where you want to save your reports.
Click new to create a new report.



CSV Attribute Names.rtb
CSV Column Names.rtb
CSV Domain Names.rtb
CSV Entity Names.rtb
CSV Table Names.rtb
demo.rtb
HTML Entity Primary Key.rtb
HTML Physical Only Columns.rtb
HTML Table Owner.rtb
REPORTS.ERP
RTF Entity Primary Key.rtb
RTF Physical Only Columns.rtb
RTF Relationship Report.rtb
Standard.erp

Directory: C:\Program Files\Computer Associates\ERwin

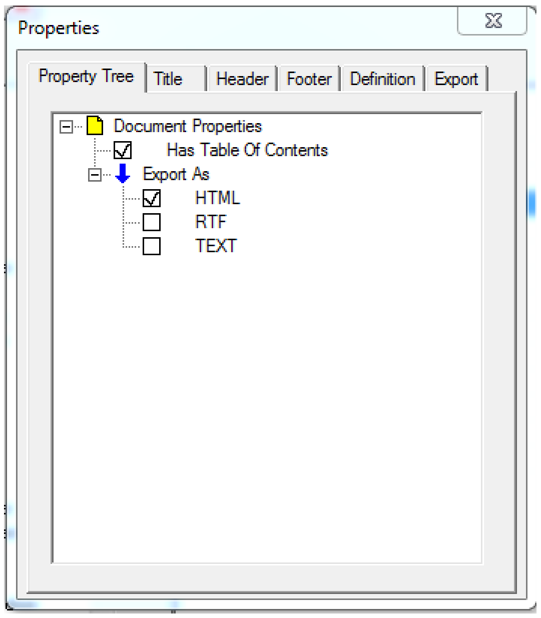
The dialog box as shown in figure 12 will pop up.



Document Untitled
Document Properties
Has Table Of Contents
Export As
HTML

Figure 12: report template builder

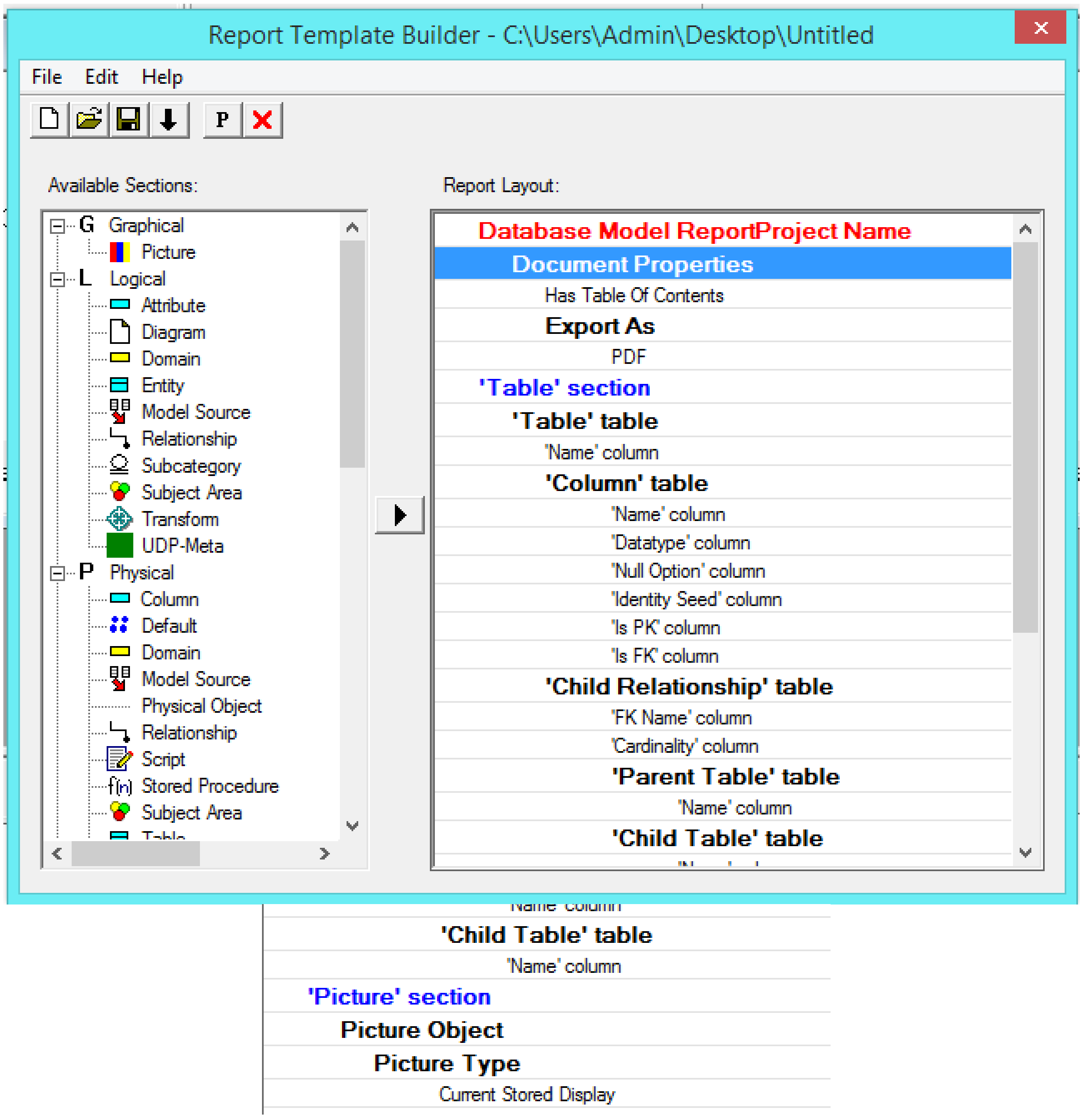
You can select various options to build a report template which, can be explored in detail in the Erwin guide provided. You can export the document as HTML, RTF or text.


Properties

Property Tree | Title | Header | Footer | Definition | Export

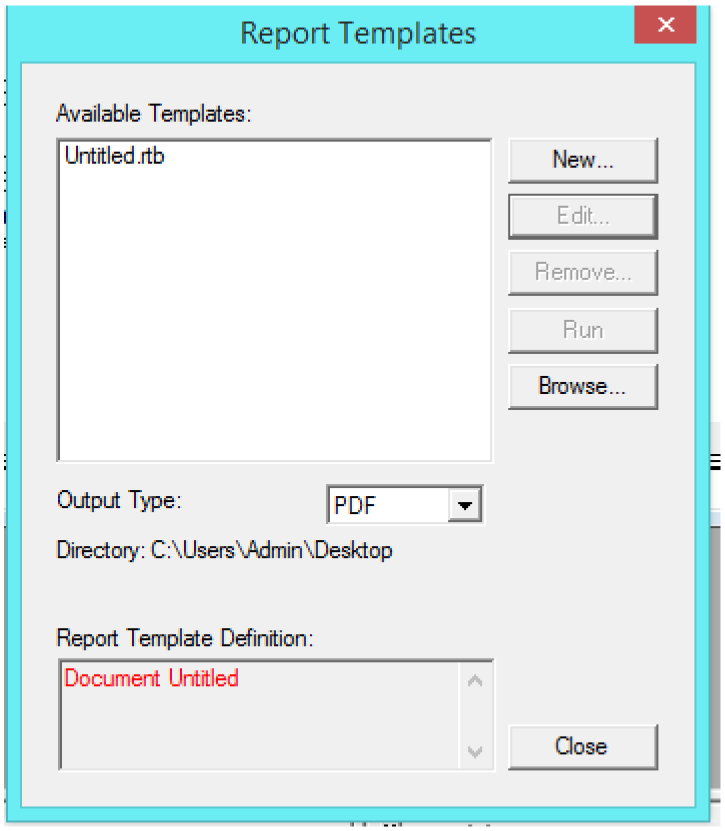
- Document Properties
  - [✓] Has Table Of Contents
  - Export As
    - [✓] HTML
    - [ ] RTF
    - [ ] TEXT

Select the Entity section from the right panel. We created two entities in the last section. We want to generate the data dictionaries for Data Model we created in the last section. Select the table and diagram sections from the right panel as shown in figure below.



Database Model ReportProject Name
Document Properties
Has Table Of Contents
Export As
PDF
'Table' section
'Table' table
'Name' column
'Column' table
'Name' column
'Datatype' column
'Null Option' column
'Identity Seed' column
'Is PK' column
'Is FK' column
'Child Relationship' table
'FK Name' column
'Cardinality' column
'Parent Table' table
'Name' column
'Child Table' table
'Name' column
'Picture' section
Picture Object
Picture Type
Current Stored Display

Now, Save your report template. Close the report template builder window. Select the template you just created and click run to generate the data dictionary as Shown in the figure below.



Select Output tyev as PDF. After clicking run, the generated report will get displayed as shown in file Reprt.PDF .

References

- http://www.isqa.unomaha.edu/wolcott/tutorials/erwin/erwin.html
- http://stackoverflow.com/questions/762937/whats-the-difference-between-identifying-and-non-identifying-relationships

InLab Question:

Case Study:

Problem Statement:

The case study titled Library Management System is library management software for the purpose of monitoring and controlling the transactions in a library. This case study on the library management system gives us the complete information about the library and the daily transactions done in a Library. We need to maintain the record of new s and retrieve the details of books available in the library which mainly focuses on basic operations in a library like adding new member, new books, and up new information, searching books and members and facility to borrow and return books. It features a familiar and well thought-out, an attractive user interface, combined with strong searching, insertion and reporting capabilities. The report generation facility of library system helps to get a good idea of which are ths borrowed by the members, makes users possible to generate hard copy.

The following are the brief description on the functions achieved through this case study:

End-Users:

•Librarian: To maintain and update the records and also to cater the needs of the users.
•Reader: Need books to read and also places various requests to the librarian.
•Vendor: To provide and meet the requirement of the prescribed books.