CHAPTER 1

ER Modeling Tools

This chapter introduces ERWin, a popular data-modeling tool used in the industry. ERWin is a powerful tool that allows database designers to enter their Entity Relationship (ER) diagrams in a graphical form and produce physical database designs for popular relational database management systems such as Oracle and Microsoft SQLServer.

The use of ERWin is illustrated in this chapter using the ER schema diagram for the COMPANY database shown in Figure 7.2 of the Elmasri/Navathe text.

1.1 Starting with ERWin

The ERWin Data Modeler workspace is shown in Figure 1.1.

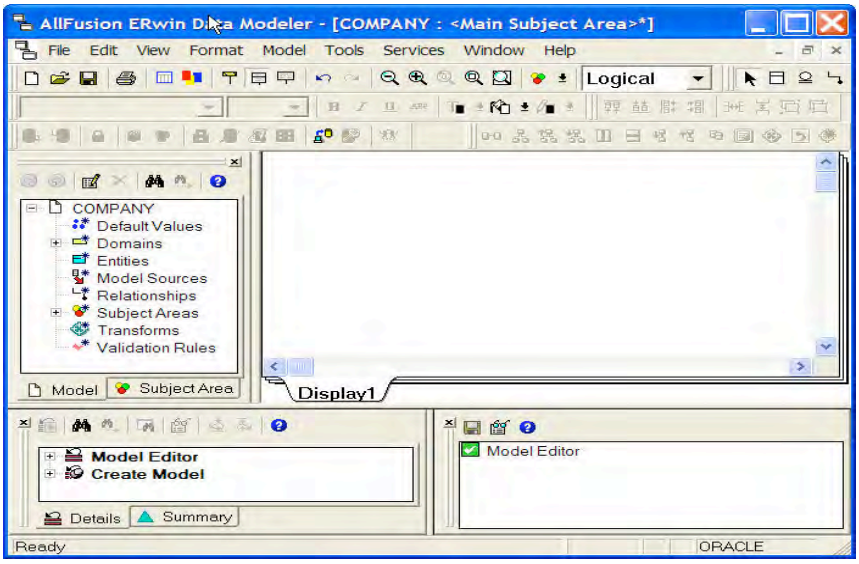

Figure 1.1: ERWin Data Modeler Workspace

The top part of the workspace consists of Menu and Toolbars. The middle part of the workspace consists of two panes: the *model explorer panel* on the left providing a text based view of the data model and the *diagram window panel* on the right providing a graphical view of the data model. The lower part of the workspace consists of two panes: the *action log panel* on the left that displays a log of all changes made to the data model under design and the *advisories panel* that displays messages associated with the actions performed on the data model under design.

ERWin supports three model types for use by the database designer:

1. Logical: A conceptual model that includes entities, relationships, and attributes. This model type is essentially at the ER modeling level.

2. Physical: A database specific model that contains relational tables, columns and associated data types.
3. Logical/Physical: A single model that includes both the conceptual level objects as well as physical level tables. In this chapter we will use this model type.

To create a model in ERWin, one should launch the program and then choose the "New" option from the File menu. The Create Model dialog appears as shown in Figure 1.2.

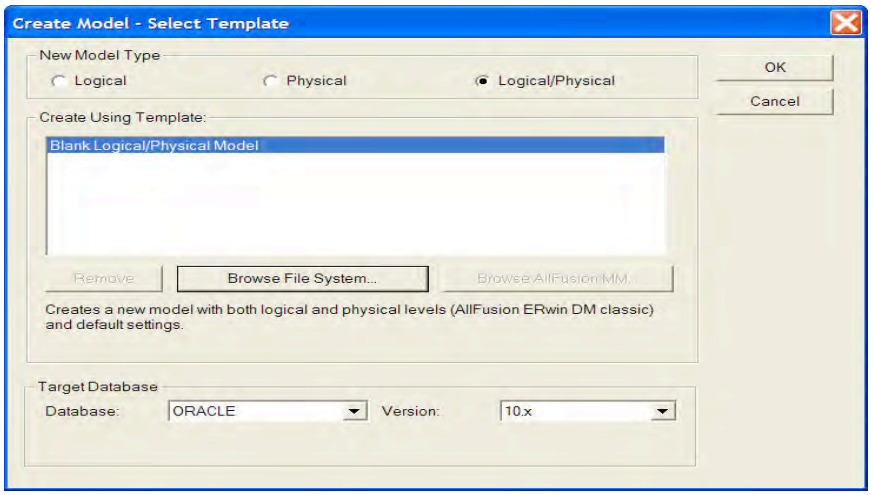

Figure 1.2: Create Model dialog window

In this dialog window, the user should choose the type of model. Typically the Logical/Physical model type should be chosen if the final goal is to produce a relational design for the database. The target database may also be chosen. In this case, Oracle 10.x version is chosen as the target database. In a future step, we will illustrate how ERWin can be used to generate SQL code to create the database objects in Oracle 10.x database.

The workspace for the new model will be populated by the system with a default name of Model_n. This name may be changed in the model explorer pane by right clicking the model name and choosing the Properties option. This brings up a new window in which the name and other properties of the model may be changed. Besides changing the model name, the "Transform" options should be checked. This would allow for many-to-many relationships to be transformed correctly into separate relational tables in the physical model. In addition any sub-type/super-type relationships will also be transformed correctly in the physical model.

1.2 Adding Entity Types

To add an entity type to the database design, the user may either right click the "Entities" entry in the model explorer pane and choose "New" or choose the "Entity" icon in the Menus and Toolbars section of the workspace and click in the diagram window panel. An entity box shows up in the diagram window panel with a default entity name (E/n) that can be changed either in the diagram window panel or in the model explorer pane. Figure 1.3 shows the addition of the EMPLOYEE entity type in the COMPANY database.

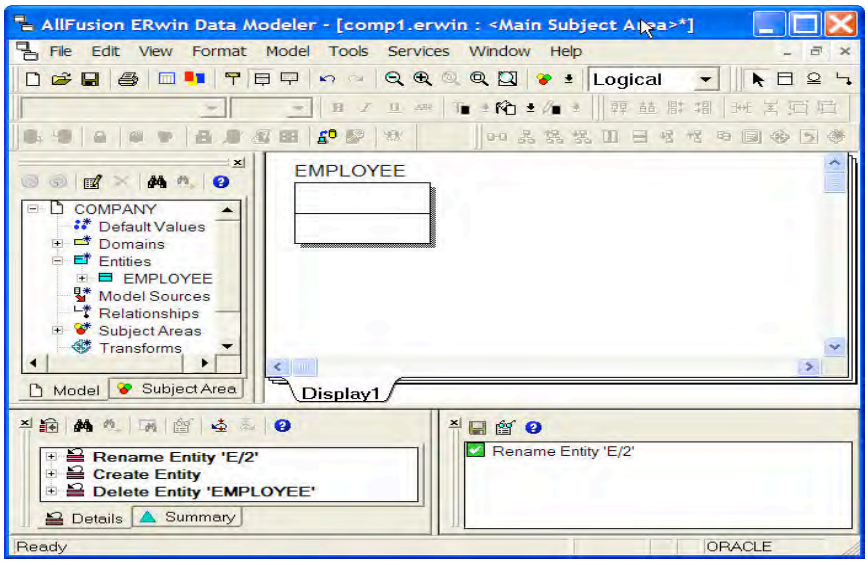

Figure 1.3: Add EMPLOYEE entity to the COMPANY database

To add attributes to the EMPLOYEE entity type, the user may right click within the EMPLOYEE entity box in the diagram window panel and choose "Attributes". This brings up a separate window using which new attributes may be added. The attribute window is shown in Figure 1.4.

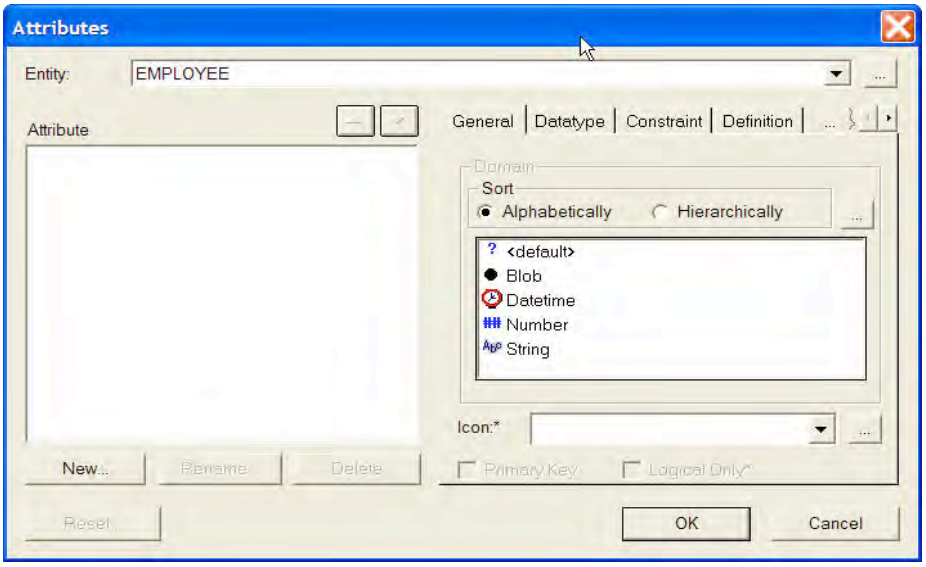

Figure 1.4: Attribute Window

The user may now add attributes one at a time by clicking the "New" button. A separate window pops up as shown in Figure 1.5.

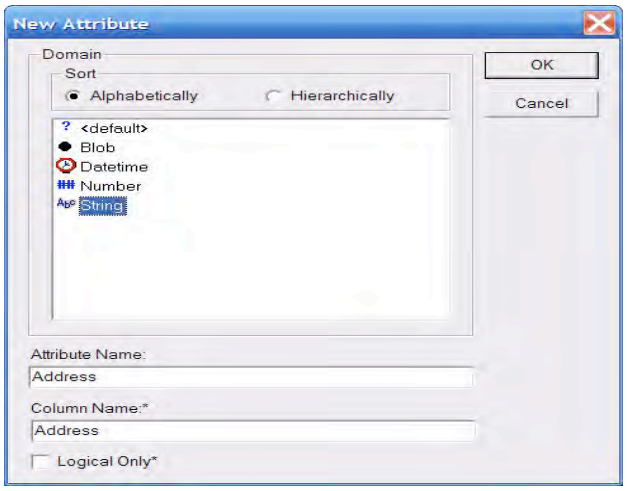

Figure 1.5: New Attribute Window

The user may choose an appropriate Domain (data type) and enter the Attribute Name and click OK. The data type may be further refined in the Attribute Window by choosing the Datatype tab and entering a precise data type. The user may also choose to designate this attribute as a primary key by selecting this option in the Attribute window.

After adding a few attributes to the EMPLOYEE entity type the Attributes window is shown in Figure 1.6.

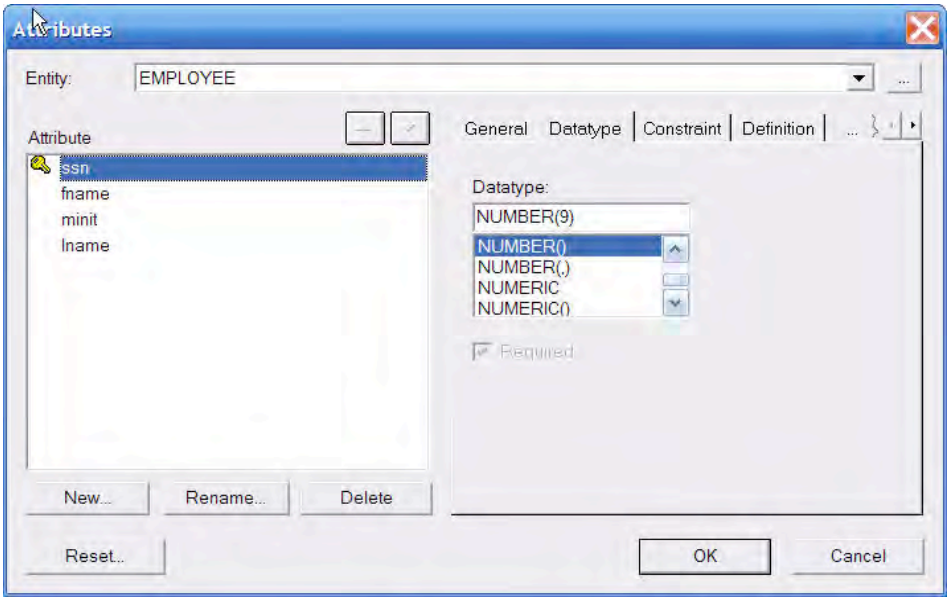

Figure 1.6: Attribute Window with four attributes

In this way, we can create each of entity types: EMPLOYEE, DEPARTMENT, PROJECT, and DEPENDENT for the COMPANY database.

Weak Entity Sets

By default any entity type created as discussed so far is classified as an independent entity type. ERWin will classify an entity type as "weak" as soon as it participates in an identifying relationship. For example, the entity type DEPENDENT will be classified as "weak" in a subsequent step when we add the identifying relationship from EMPLOYEE to DEPENDENT in the next section. Weak entity types are denoted by rounded rectangles in the diagram window panel.

Multi-Valued Attributes

Multi-valued attributes such as the locations attribute for the DEPARTMENT entity type cannot be modeled easily with ERWin. To handle such attributes, a separate entity type LOCATIONS is created and a many-to-many relationship between DEPARTMENT and LOCATIONS will be established in the next section.

1.3 Adding Relationships

Three types of relationships are supported in ERWin: identifying, non-identifying, and many-to-many. ERWin classifies the child entity type in an identifying relationship as "weak". To add a relationship, the user may simply right click the Relationships entry in the model explorer pane and choose "New". This pops up a new relationship window as shown in Figure 1.7.

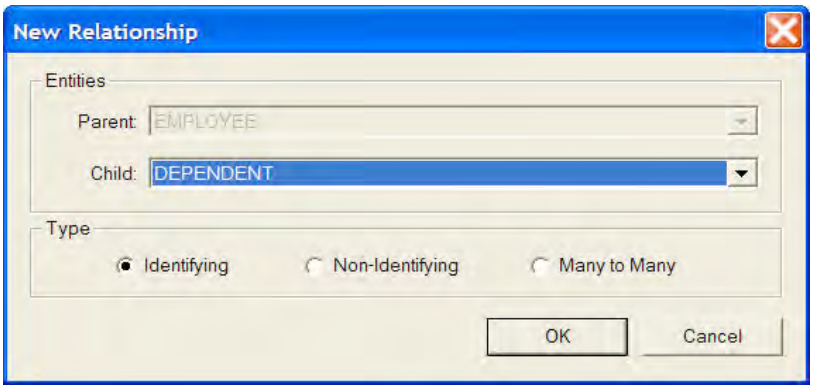

Figure 1.7: New Relationship Window

After choosing the parent and child entity types and the type of relationship and clicking OK, the new relationship is added and is reflected by a line connecting the two entity types in the diagram window panel. The many-to-many relationships are denoted by solid connecting lines, with two black dots at the two ends. Non-identifying relationships are denoted by a dashed connecting line with a black dot at many-end and a square-shaped symbol at the one-end. Identifying relationships are denoted by a solid connecting line with a black dot at the many-end and nothing special at the one-end.

After creating a new relationship, the user may add verb phrases and other properties of the relationship by right clicking the connecting line in the diagram and choosing properties.

In the case of the COMPANY database, we create the following relationships:

- One identifying relationship from EMPLOYEE to DEPENDENT.
- Two many-to-many relationships, one from EMPLOYEE to PROJECT and the other from DEPARTMENTS to LOCATIONS, and
- Four non-identifying relationships: from EMPLOYEE to DEPARTMENT (one-to-one for manages), from DEPARTMENT to EMPLOYEE (one-to-many for works for relationship), from EMPLOYEE to EMPLOYEE (one-to-many for supervisor/supervisee relationship), and from DEPARTMENT to PROJECT (one-to-many for the controls relationship).

The final logical ER diagram from the diagram window panel is shown in Figure 1.8.

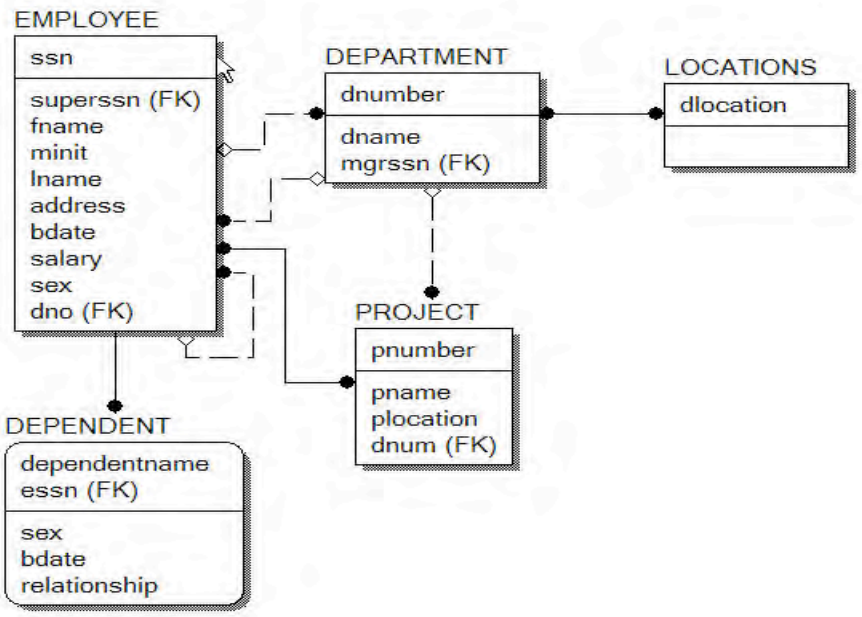

Figure 1.8: Final Logical ER Diagram

Notice that the two many-to-many relationships do not have the transforms applied yet. The transforms are shown in the physical ER diagram (obtained by switching from Logical to Physical in the Menu and Toolbar section) in Figure 1.9. Notice the introduction of the two new "entity types" for the two many-to-many relationships. These entity types are introduced because the transforms are defined at the model level.

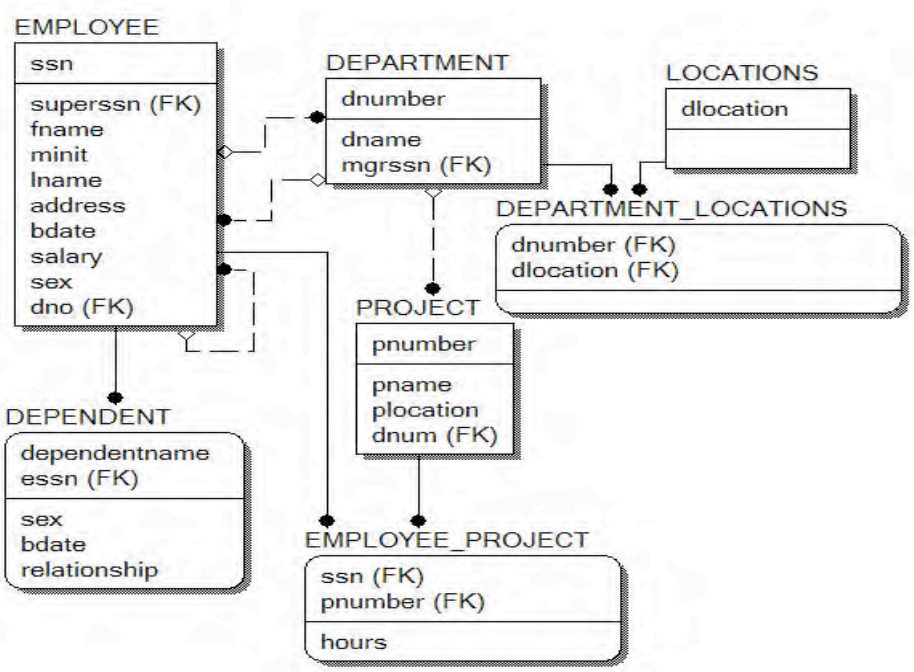

Figure 1.9: Final Physical ER Diagram

1.4 Forward Engineering

ERWin provides a powerful feature called forward engineering that allows the database designer to convert the ER design into a schema generation SQL script for one or more target relational databases. The following SQL script is obtained for the COMPANY database by choosing Tools→Forward Engineering→Schema-Generation option in the Menus and Toolbars section and clicking the "Preview" button.

```
CREATE TABLE DEPARTMENT
(
      dname VARCHAR2(20)  NOT NULL ,
      dnumber INTEGER  NOT NULL ,
      mgrssn NUMBER(9)  NULL
);

ALTER TABLE DEPARTMENT
      ADD PRIMARY KEY (dnumber);

CREATE TABLE DEPARTMENT_LOCATIONS
(
      dnumber  INTEGER  NOT NULL ,
      dlocation  VARCHAR2(20)  NOT NULL
);


ALTER TABLE DEPARTMENT_LOCATIONS
      ADD  PRIMARY KEY (dnumber,dlocation);
```

```
CREATE TABLE DEPENDENT
(
      dependentname  VARCHAR2(20)  NOT NULL ,
      sex  CHAR  NULL ,
      bdate  DATE  NULL ,
      relationship  VARCHAR2(20)  NULL ,
      essn  NUMBER(9)  NOT NULL
);


ALTER TABLE DEPENDENT
      ADD  PRIMARY KEY (dependentname,essn);


CREATE TABLE EMPLOYEE
(
      ssn  NUMBER(9)  NOT NULL ,
      superssn  NUMBER(9)  NULL ,
      fname  VARCHAR2(20)  NULL ,
      minit  CHAR  NULL ,
      lname  VARCHAR2(20)  NOT NULL ,
      address  VARCHAR2(50)  NULL ,
      bdate  DATE  NULL ,
      salary  NUMBER(8)  NULL ,
      sex  CHAR  NULL ,
      dno  INTEGER  NULL
);


ALTER TABLE EMPLOYEE
      ADD  PRIMARY KEY (ssn);


CREATE TABLE EMPLOYEE_PROJECT
(
      ssn  NUMBER(9)  NOT NULL ,
      pnumber  INTEGER  NOT NULL ,
      hours  NUMBER(3)  NULL
);


ALTER TABLE EMPLOYEE_PROJECT
      ADD  PRIMARY KEY (ssn,pnumber);


CREATE TABLE LOCATIONS
(
      dlocation  VARCHAR2(20)  NOT NULL
);


ALTER TABLE LOCATIONS
      ADD  PRIMARY KEY (dlocation);
```

```
CREATE TABLE PROJECT
(
      pnumber  INTEGER  NOT NULL ,
      pname  VARCHAR2(20)  NULL ,
      plocation  VARCHAR2(20)  NULL ,
      dnum  INTEGER  NULL
);


ALTER TABLE PROJECT
      ADD  PRIMARY KEY (pnumber);


ALTER TABLE DEPARTMENT
      ADD ( FOREIGN KEY (mgrssn) REFERENCES EMPLOYEE(ssn) ON DELETE SET NULL);


ALTER TABLE DEPARTMENT_LOCATIONS
      ADD ( FOREIGN KEY (dnumber) REFERENCES DEPARTMENT(dnumber));


ALTER TABLE DEPARTMENT_LOCATIONS
      ADD ( FOREIGN KEY (dlocation) REFERENCES LOCATIONS(dlocation));


ALTER TABLE DEPENDENT
      ADD ( FOREIGN KEY (essn) REFERENCES EMPLOYEE(ssn));


ALTER TABLE EMPLOYEE
      ADD  ( FOREIGN  KEY  (superssn)  REFERENCES  EMPLOYEE(ssn)  ON  DELETE  SET
NULL);


ALTER TABLE EMPLOYEE
      ADD  ( FOREIGN  KEY  (dno)  REFERENCES  DEPARTMENT(dnumber)  ON  DELETE  SET
NULL);


ALTER TABLE EMPLOYEE_PROJECT
      ADD ( FOREIGN KEY (ssn) REFERENCES EMPLOYEE(ssn));


ALTER TABLE EMPLOYEE_PROJECT
      ADD ( FOREIGN KEY (pnumber) REFERENCES PROJECT(pnumber));


ALTER TABLE PROJECT
      ADD  ( FOREIGN  KEY  (dnum)  REFERENCES  DEPARTMENT(dnumber)  ON  DELETE  SET
NULL);
```

The above SQL script contains table definitions and basic primary and foreign key constraints definitions. ERWin does provide a number of options to generate views, triggers, indices etc. and these can be set in the forward engineering schema generation window.

1.5 Supertype/Subtype Example

ERWin supports the creation of sub-type/super-type relationships between entity types. Consider the example in Figure 8.3 of the Elmasri/Navathe text in which a super-type entity VEHICLE consists of two sub-types CAR and TRUCK. To create this design in ERWin, the three entity types are created first. Then, the user may click the sub-type button (a circle with two parallel lines below the circle) in the Menus and Toolbars section, followed by clicking the super-type entity (VEHICLES) in the diagram window pane, followed by clicking the sub-type entity (CAR) in the diagram window pane. This process may be repeated for adding other sub-types (TRUCK in this example). The logical model for this example is shown in Figure 1.10.

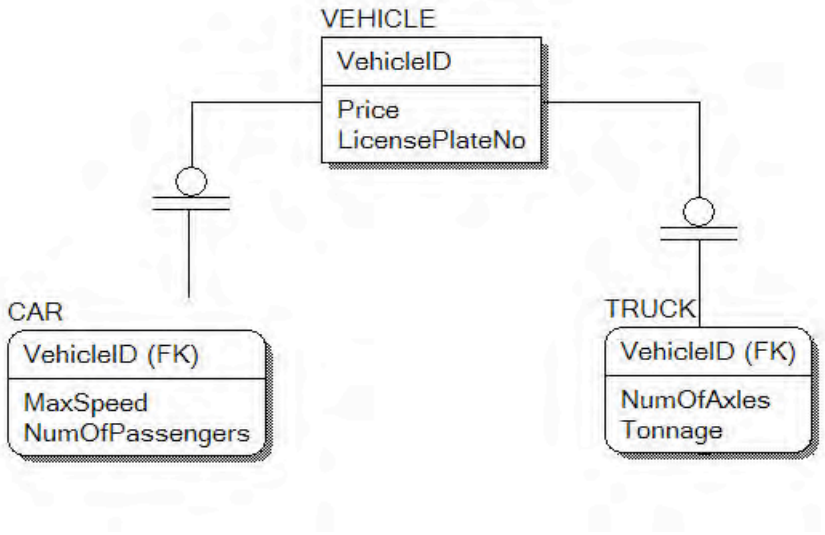

Figure 1.10: Sub-type/Super-type Logical ER Diagram

To customize the properties of the sub-type/super-type relationship, the user may right click the relationship symbol (circle with two parallel lines) and choose Subtype Relationship. This brings up a window shown in Figure 1.11. The user may choose "Complete" subtype (when all categories are known) or "Incomplete" subtype (when all categories may not be known). The user may also add verb phrases etc by right-clicking the relationship line and choosing properties as was done for ordinary relationships. ERWin also allows the user to choose a "discriminator" attribute for the sub-types (an attribute in the super-type whose values determine the sub-type object). If no discriminator attribute is defined, the user may choose "-- -- --".

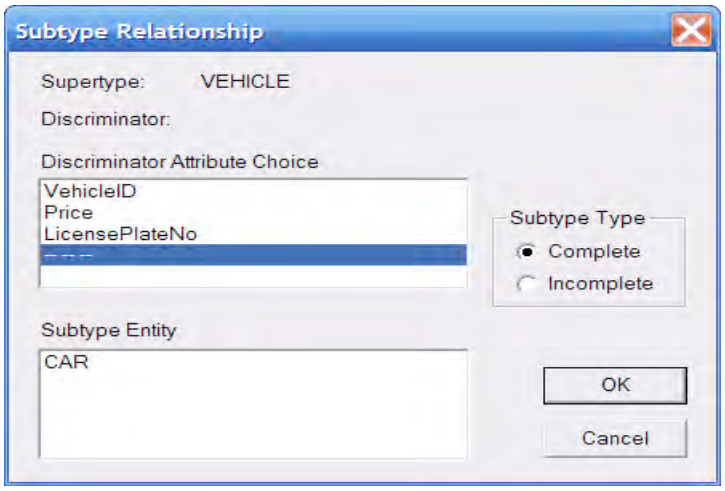

Figure 1.11: Subtype Relationship Properties

The following SQL script is produced using the forward engineering feature of ERWin for the Vehicles example:

```
CREATE TABLE CAR
(
      MaxSpeed  INTEGER  NULL ,
      NumOfPassengers  INTEGER  NULL ,
      VehicleID  INTEGER  NOT NULL
);


ALTER TABLE CAR
      ADD  PRIMARY KEY (VehicleID);


CREATE TABLE TRUCK
(
      NumOfAxles  INTEGER  NULL ,
      Tonnage  INTEGER  NULL ,
      VehicleID  INTEGER  NOT NULL
);


ALTER TABLE TRUCK
      ADD  PRIMARY KEY (VehicleID);


CREATE TABLE VEHICLE
(
      VehicleID  INTEGER  NOT NULL ,
      Price  NUMBER(8,2)  NULL ,
      LicensePlateNo  VARCHAR2(20)  NULL
);


ALTER TABLE VEHICLE
      ADD  PRIMARY KEY (VehicleID);
```

```
ALTER TABLE CAR
      ADD ( FOREIGN KEY (VehicleID) REFERENCES VEHICLE(VehicleID));


ALTER TABLE TRUCK
      ADD ( FOREIGN KEY (VehicleID) REFERENCES VEHICLE(VehicleID));
```

Exercises

ER Modeling Problems

1. Consider the *university* database described in Exercise 7.16 of the Elmasri/Navathe text. Enter the ER schema for this database using a data-modeling tool such as ERWin.

2. Consider a *mail order* database in which employees take orders for parts from customers. The data requirements are summarized as follows:

   - The mail order company has employees identified by a unique employee number, their first and last names, and a zip code where they are located.
   - Customers of the company are uniquely identified by a customer number. In addition, their first and last names and a zip code where they are located are recorded.
   - The parts being sold by the company are identified by a unique part number. In addirion, a part name, their price, and quantity in stock are recorded.
   - Orders placed by customers are taken by employees and are given a unique order number. Each order may contain certain quantities of one or more parts and their received date as well as a shipped date is recorded.

   Design an Entity-Relationship diagram for the mail order database and enter the design using a data-modeling tool such as ERWin.

3. Consider a *movie* database in which data is recorded about the movie industry. The data requirements are summarized as follows:
   - Movies are identified by their title and year of release. They have a length in minutes. They also have a studio that produces the movie and are classified under one or more genres (such as horror, action, drama etc). Movies are directed by one or more directors and have one or more actors acting in them. The movie also has a plot outline. Each movie also has zero or more quotable quotes that are spoken by a particular actor acting in the movie.
   - Actors are identified by their names and date of birth and act in one or more movies. Each actor has a role in the movie.
   - Directors are also identified by their names and date of birth and direct one or more movies. It is possible for a director to act in a movie (not necessarily in a movie they direct).

- Studios are identified by their names and have an address. They produce one or more movies.

Design an Entity-Relationship diagram for the movie order database and enter the design using a data-modeling tool such as ERWin.

4. Consider a *conference review* system database in which researchers submit their research papers for consideration. The database system also caters to reviewers of papers who make recommendations on whether to accept or reject the paper. The data requirements are summarized as follows:
   Authors of papers are uniquely identified by their email id. Their first and last names are also recorded.
   - Papers are assigned unique identifiers by the system and are described by a title, an abstract, and a file name containing the actual paper.
   - Papers may have multiple authors, but one of the authors is designated as the contact author.
   - Reviewers of papers are uniquely identified by their email id. Their first and last names are also recorded.
   - Each paper is assigned between two and four reviewers. The reviewer rates the paper assigned to him on a scale of 1 to 10.
   - Each review contains two types of written comments: one to be seen by the review committee only and the other by the author(s) as well.

   Design an Entity-Relationship diagram for the conference review database and enter the design using a data-modeling tool such as ERWin.

5. Consider the ER diagram for the AIRLINE database shown in Figure 7.20 of the Elmasri/Navathe text. Enter this design using a data-modeling tool such as ERWin.

Enhanced ER Modeling Problems

6. Consider a *grade book* database in which instructors within an academic department maintain scores/points obtained by individual students in their classes. The data requirements are summarized as follows:

   - Students are identified by a unique student id, their first and last names, and an email address.
   - The instructor teaches certain courses each term. The courses are uniquely identified by a course number, a section number, and the term in which they are taught. The instructor also assigns grade cutoffs (example 90, 80, 70, and 60) for letter grades A, B, C, D, and F for each course he or she teaches.
   - Students are enrolled in courses taught by the instructor.
   - Each course being taught by the instructor has a number of grading components (such as mid-term, final exam, project, etc.). Each grading component has a maximum number of points (such as 100 or 50) and a weight (such as 20% or 10%). The weights of all the grading components of a course usually add up to 100.

- Finally, the instructor records the points earned by each student in each of the grading components in each of the courses. For example, student with id=1234 earns 84 points for the grading component mid-term for the course CSc 2310 section 2 in the fall 2005 term. The mid-term grading component may have been defined to have a maximum of 100 points and a weight of 20% of the course grade.

Design an enhanced Entity-Relationship diagram for the grade book database and enter the design using a data-modeling tool such as ERWin.

7. Consider an *online auction* database system in which members (buyers and sellers) participate in the sale of items. The data requirements for this system are summarized as follows:

   - The online site has members who are identified by a unique member id and are described by an email address, their name, a password, their home address, and a phone number.
   - A member may be a buyer or a seller. A buyer has a shipping address recorded in the database. A seller has a bank account number and routing number recorded in the database.
   - Items are placed by a seller for sale and are identified by a unique item number assigned by the system. Items are also described by an item title, an item description, a starting bid price, bidding increment, the start date of the auction, and the end date of the auction.
   - Items are also categorized based on a fixed classification hierarchy (for example a modem may be classified as /COMPUTER/HARDWARE/MODEM).
   - Buyers make bids for items they are interested in. A bidding price and time of bid placement is recorded. The person at the end of the auction with the highest bid price is declared the winner and a transaction between the buyer and the seller may proceed soon after.
   - Buyers and sellers may place feedback ratings on the purchase or sale of an item. The feedback contains a rating between 1 and 10 and a comment. Note that the rating is placed by the buyer or seller involved in the completed transaction.

   Design an Entity-Relationship diagram for the auction database and enter the design using a data-modeling tool such as ERWin.

8. Consider a database system for a baseball organization such as the major leagues. The data requirements are summarized as follows:

   - The personnel involved in the league include players, coaches, managers, and umpires. Each is identified by a unique personnel id. They are also described by their first and last names along with the date and place of birth.
   - Players are further described by other attributes such as their batting orientation (left, right, or switch) and have a lifetime batting average (BA).
   - Within the players group is a subset of players called pitchers. Pitchers have a life-time ERA (earned run average) associated with them.

- Teams are uniquely identified by their names. Teams are also described by the city in which they are located and the division and league in which they play (such as Central division of the American league).
- Teams have one manager, a number of coaches, and a number of players.
- Games are played between two teams with one designated as the home team and the other the visiting team on a particular date. The score (runs, hits, and errors) are recorded for each team. The team with more number of runs is declared the winner of the game.
- With each finished game, a winning pitcher and a losing pitcher are recorded. In case there is a save awarded, the save pitcher is also recorded.
- With each finished game, the number of hits (singles, doubles, triples, and home runs) obtained by each player is also recorded.

Design an Entity-Relationship diagram for the baseball database and enter the design using a data-modeling tool such as ERWin.

9. Consider the ER diagram for the university database shown in Figure 8.9 of the Elmasri/Navathe text. Enter this design using a data-modeling tool such as ERWin.

10. Consider the ER diagram for the small airport database shown in Figure 8.12 of the Elmasri/Navathe text. Enter this design using a data-modeling tool such as ERWin.