Answers to Review Questions

1. Name and discuss the different levels of data abstraction as defined by ANSI/SPARC.

**Conceptual**

Located at the abstraction apex, the conceptual model represents a global view of the data. It forms the basis of the conceptual schema, which provides a relatively easily understood bird's-eye view of the data environment. Since the conceptual model focuses on the main data objects and avoids detail, it exhibits both software and hardware independence. The most widely used conceptual model is the Entity Relationship (E-R) model, which yields the basic database blueprint.

**Internal**

The internal model represents the adaptation of the conceptual model to a specific DBMS. Basically, the internal model requires the database designer to match the conceptual model's characteristics and constraints to those of the selected hierarchical, network, or relational DBMS. Therefore, although it is still hardware-independent, it is software-dependent.

**External**

The external model represents the applications programmer's view of the data environment. Its use requires that the modeler subdivide a universal set of requirements and constraints into functional modules, each represented by its own external model. (The modules correspond to business units such as production, sales, personnel, and so on.) Each external model is represented by its own external schema. Each business unit is thus represented by an external model that includes that unit's entities, the relationships between the entities, and its constraints. Since external models are defined for a specific DBMS, they are DBMS (software)-dependent, but hardware-independent.

The use of external schemas has several important advantages:

- Using database subsets makes it easier to view the specific application program requirements.
- The subsets make it easier to identify specific data required to support a given business unit's operations.
- They make it easier to examine feedback about the conceptual model's adequacy.
- It is more difficult to damage the entire database when each module employs only the required data subset.
Physical

The physical model operates at the lowest level of abstraction. It describes the way data will be saved on storage media such as disks or tapes. Because the physical model requires the specification of physical storage devices and the access methods required to reach the data located on those storage devices, the physical model is both hardware-and software-dependent. (See section 3.2.4.)

2. What are the main building modules of the Entity Relationship model? Discuss each one.

The E-R model contains the following components:

**Entities** are used to represent real world objects such as professor, class, student, etc. Actually, the E-R model uses entity *sets*, which are the grouping of related entities. For example, the entity set named STUDENT is composed of specific entities such as Anne R. Morowski, William F. Achero, and so on. So, when E-R models are used, the word entity actually refers to entity set.

Entity types include *weak* and *composite* entities. Weak entities have two characteristics:

1. They are existence-dependent, i.e., they cannot exist without some other entity. (Example: DEPENDENT cannot exist without EMPLOYEE.)

2. The weak entity's primary key is partially or wholly derived from the parent side of the relationship.

Composite entities are used to transform M:N relationships into sets of 1:M relationships. The composite entity's primary key consists of the combination of primary keys from the entities it connects.

Finally, entities may be classified as supertypes and subtypes. In effect, we can create a generalization hierarchy to represent entities that share common characteristics. For example, all EMPLOYEE entities share attributes such as NAME, ADDRESS, HOME_PHONE, etc. However, special employees such as pilots have attributes, such as FLT_TIME, not shared by other employees. Thus PILOT becomes a subtype of the supertype EMPLOYEE. (See the detailed discussion in section 3.3.12!)

**Attributes** are the characteristics that define an entity. For example, the attributes NAME, ADDRESS, PHONE, and MAJOR are some of the characteristics of the STUDENT. Some attributes perform the crucial role of identifying each entity uniquely. Such attributes form the entity's primary key.
Attributes are classified as **simple** or **composite**. A composite attribute is one that can be subdivided to yield additional attributes. For example, a phone number can be subdivided into area code, exchange, and number. A simple attribute cannot be subdivided.

Attributes may also be **single-valued** or **multivalued**. As the name suggests, a single-valued attribute can have only one value. For example, a person can have only one gender and one birth date. Multivalued attributes can have many values: A person can have more than one college degree, and a household may have more than one phone.

**Relationships** are associations between entities. Thus the relationship between the entities **PROFESSOR** and **CLASS** is **teaches**, i.e., a **PROFESSOR** **teaches** a **CLASS**.

Relationships are classified by their degree. Unary relationships are those whose entities form relationships with themselves, thus producing so-called recursive relationships. For example, a **COURSE** may be a **prerequisite** to a **COURSE**.

Binary relationships, i.e., relationships between two different entities, are most common and form the basis for most E-R modeling work. For example, a **PILOT** **flies** an **AIRCRAFT**.

Ternary relationships are those found among three different entities. For example, a **CONTRIBUTOR** **contributes** to a **FUND**, and a **RECIPIENT** **receives funding** from that **FUND**. Higher-order relationships than ternary are uncommon in E-R modeling.

Relationships are also classified as 1:1, 1:M, and M:N. The term **connectivity** is used to describe such classifications. The participation of relationships may also be classified as optional or mandatory.

**3. What is a composite entity, and when is it used?**

A composite entity, also known as a bridge entity, is one that is used to transform M:N relationships into sets of 1:M relationships. The composite entity's primary key consists of the combination of primary keys from the entities it connects. For a detailed review of the role played by composite entities, refer to the second discussion focus question (**How are M:N relationships handled in the development of an E-R diagram?**)

**4. Why is data modeling considered a "communication tool"?**

Data are the basic information units used by any system. The problem is that different groups of data users view the data from a different perspective. This perspective is based on a host of factors: Clerks or sales personnel have different data needs from the purchasing manager and the company president. Application programmers and developers have yet different views. To use the analogy presented in the book, different groups of information users and producers tend to reflect the "blind men and the elephant" allegory: The one who felt the elephant's tail had quite a different view of the elephant from the one who felt its trunk. What we need is the ability to see the whole
elephant. Similarly, a house is not a random collection of rooms; if we intend to build a house, we should first have an overall view provided by its blueprint. Similarly, a sound data environment requires the overall view of the data provided by the database blueprint, which is based on a data model.

To serve as the vehicle for database blueprinting, a good data model helps us understand the complexities of the real-world environment. Such understanding facilitates communication between the various data users and producers. The use of the blueprint analogy is appropriate because, like any building blueprint, the database blueprint based on a data model enables us to see what data are needed, how the data are to be stored, by whom the data are to be used, and how the data are related. A good data model ultimately enables us to understand how an organization functions, thus enabling us to communicate the hows and whats to that organization's data users and producers.

5. Suppose you are working within the framework of the conceptual model in Figure Q3.5:

![External model #1 and External model #2](image)

Figure Q3.5 The Conceptual Model for Question 5

Given the conceptual model in Figure Q3.5, create two external models.
6. How are database models related to the level of data abstraction? (*Hint:* Why do database designers find it so much easier to develop designs based on the relational model?)

ANSI/SPARC defines four different data models (conceptual, internal, external, and physical) based on three levels of abstraction: high, medium, and low. To a greater or lesser extent, these data models are implemented through a hierarchical, network, or relational DBMS. (See Figure 3.1 in the text.)

The higher the level of abstraction, the less need for the often difficult-to-understand physical details.

The hierarchical and network database models make use of all three levels of abstraction. In contrast, the relational model primarily makes use of high and medium degrees of abstraction. In fact, the relational model generally allows its users to focus on the highest and, therefore, easiest to understand, level of abstraction. Given this focus, the relational implementation model lets us make better use of the conceptual data model. The conceptual model, being located at the abstraction apex, identifies and describes only the main data objects, avoiding the details. And it is this conceptual model that underlies the widely used E-R model whose use produces valuable database blueprints.

7. How would you (graphically) identify each of the following E-R model components?

a. An entity

An entity is represented by a rectangle containing the entity name. (Remember that, in E-R modeling, the word "entity" actually refers to the entity set.) A weak entity is indicated through the use of a double-line rectangle. A composite entity is identified through a diamond within a rectangle.

b. An attribute

An attribute is identified by an oval and is connected to the entity through a:

- single line to indicate a single-valued, non-derived attribute
- dotted line to indicate a derived attribute
- double line to indicate multivalued attributes
- The attribute oval contains the attribute name. A primary key attribute is underlined.
Keep in mind that such attribute depictions clutter the E-R presentations and they are, therefore, usually omitted.

c. A relationship

A relationship is indicated by a diamond-shaped symbol located between the entities whose relationship is being described. The diamond contains the relationship description, written in lowercase letters. The relationship description is usually an active verb such as teaches, flies, writes, etc. However, the relationship may warrant a more passive description to improve the E-R diagram's readability for humans: PILOT is an EMPLOYEE, INVOICE_LINE is contained within INVOICE, etc.

Relationships are implemented through the use of foreign keys. Review the example given in discussion question 2. (How are M:N relationships handled in the development of an E-R diagram?) Also review the many models and their tables presented in the chapter.

8. The Hudson Engineering Group (HEG) has contacted you to create a conceptual model whose application will meet the expected database requirements for its training program. The HEG administrator gives you the following description of the training group's operating environment:

The HEG has 12 instructors and can handle up to 30 trainees per class. HEG offers five "advanced technology" courses, each of which may generate several classes. If a class has fewer than 10 trainees in it, it will be canceled. It is, therefore, possible for a course not to generate any classes during a session. Each class is taught by one instructor. Each instructor may teach up to two classes or may be assigned to do research. Each trainee may take up to two classes per session.

Given this information, do the following:

a. Draw the E-R diagram for HEG.

b. Describe the relationship between instructor and course in terms of connectivity, cardinality, and existence dependence.

Both questions, a and b, have been addressed in the following E-R diagram. Basically, three sets of relationships exist: A COURSE may generate one or more CLASSes, an INSTRUCTOR teaches up to two CLASSes, and a TRAINEE may enroll in up to two CLASSes. A trainee can take more than one class, and each class contains many (10 or more) trainees, so there is a M:N relationship between TRAINEE and CLASS. (We must, therefore, create a composite entity to serve as the bridge between TRAINEE and CLASS.) A class is taught by only one instructor, but an instructor can teach up to two classes. Therefore, there is a 1:M relationship between INSTRUCTOR and CLASS. Finally, a COURSE may generate more than one CLASS, while each CLASS is based on
one COURSE, so there is a 1:M relationship between COURSE and CLASS. These relationships are all reflected in the following E-R diagram. Note the optional and mandatory relationships: to exist, a CLASS must have TRAINEEs enrolled in it, but TRAINEEs do not necessarily take CLASSes. (Some may take "on the job training.") An INSTRUCTOR may not be teaching any CLASSes, doing research instead, but each CLASS must have an INSTRUCTOR. If not enough people sign up for a CLASS, a COURSE may not generate any CLASSes, but each CLASS must represent a COURSE.

![E-R Diagram for HEG](image)

**Figure Q3.8 The E-R Diagram for HEG**

9. Discuss the difference between a composite key and a composite attribute. How would each be indicated in an E-R diagram?

A composite key is one that consists of more than one attribute. A composite attribute is one that can be subdivided to yield attributes for each of its components. If the E-R diagram contains the attribute names for each of its entities, a composite key is indicated in the E-R diagram by the fact that more than one attribute name is underlined to indicate its participation in the primary key. There is no E-R convention that enables us to indicate that an attribute is a composite attribute.

10. What two courses of action are available to a designer when a multivalued attribute is encountered?

The designer can split the multivalued attributes into its components and keep these components in the same entity.
Examples:

- CAR color is decomposed into TOPCOLOR, TRIMCOLOR, and BODYCOLOR.
- EMPLOYEE education is decomposed into HIGHSCHOOL, TWO-YEAR COLLEGE, FOUR-YEAR COLLEGE, and POST-GRADUATE.

The designer may also create a new entity composed of the multivalued attribute's components and link this new entity to the entity in which the multi-valued attributes occurred originally. This second option is especially desirable when the number of outcomes in the multivalued attribute is, for all practical purposes, unlimited. For example, employees classified as "technical" may have certifications in many different areas and at many different levels.

11. What is a derived attribute? Give an example.

Database designers who concentrate on designs that display "design elegance" are very reluctant to store derived attributes in the database. Instead, they prefer that these derived attribute values are computed through appropriate algorithms when they are needed in a query. For example, a person's age may be calculated by using Julian dates to subtract the birth date from the current date and dividing the resulting number of days by 365. In other words, the attribute EMP_AGE is computed by

\[
\text{EMP\_AGE} = (\text{EMP\_DOB} - \text{DATE()})/365
\]

Similarly, a sales clerk's total gross pay may be computed by adding a computed sales commission to base pay. For instance, if the sales clerk's commission is 1%, the gross pay may be computed by

\[
\text{EMP\_GROSSPAY} = \text{INV\_SALES} \times 1.01 + \text{EMP\_BASEPAY}
\]

Or the invoice line item amount may be calculated by

\[
\text{LINE\_TOTAL} = \text{LINE\_UNITS} \times \text{PROD\_PRICE}
\]

The problem with *not* storing derived attributes is that large databases tend to yield very slow queries when the derived attribute values are computed during the query. For example, if you need to get line item amounts for each product sold during a pass across ten million invoice line records, you will discover that the results are slow in coming! It is much less noticeable to the end-user when the required values are computed at the time of their generation and then to store the results in the table; the same query will execute much faster in this scenario. Here is a typical example of the trade-offs that are often required in the real world. Depending on the information requirements, the number of occurrences in the queried tables, and the frequency with which various sales revenue queries are executed, it may even be useful to store line totals in the invoice line table *AND* to store the line totals in the invoice table!
12. How is a relationship between entities indicated in an E-R diagram? Give an example, using the Chen and Crow’s foot data models.

Relationships in the CHEN model are represented by diamond-shaped symbols. The diamond is placed between the entities whose relationships are being examined. Within the diamond-shaped symbol, the relationship description is written (in lowercase letters) to express the kind of relationship that exists between the entities. Note the illustrations in Figure Q3.12.
13. What is a weak entity, and how is it represented in an E-R diagram? Give an example, using the Chen and Crow’s foot data models.

A weak entity is one that has TWO identifying characteristics:

1. It is existence-dependent on another entity, i.e., it cannot exist without the entity with which it has a relationship.

2. It inherits at least part of their primary key from the entity to which it is related.

The weak entity is indicated through the display of a double-lined box.
14. How is a composite entity represented in an E-R diagram, and what is its function? (Illustrate both the Chen and the Crow’s Foot models.)

In the Chen model, a composite entity, also known as a bridge entity, is represented through a diamond contained within a box. The diamond represents a relationship, while the box represents an entity. Thus the composite entity indicates that it translates a M:N relationship into two 1:M relationships through an entity. The label "bridge entity" is based on the fact that a composite entity serves as a connection between other entities.

The label "composite" is based on the fact that the composite entity contains at least the primary key attributes of each of the entities that are connected by it. The composite entity is an important component of the E-R model because relational database models should not contain M:N relationships.

Example:
A trucking company keeps a log of its trucking operations to keep track of its driver/truck assignments. The company may assign any given truck to any given driver many times and, as time passes, each driver may be assigned to drive many of the company's trucks:

Since this M:N relationship should not be implemented, we create the composite entity named LOG whose attributes are defined by the end-user information requirements. In this case, it may be useful to include LOG_DATE, TRUCK_NUM, DRIVER_NUM, LOG_TIME_OUT, and LOG_TIME_IN.

Note that the LOG's TRUCK_NUM and DRIVER_NUM attributes are the LOG's foreign keys that provide the bridge between the TRUCK and DRIVER, respectively. In other words, the composite entity must contain at least the primary keys of the entities connected by it. Usually, this combination of foreign keys is designated to be the composite entity's primary key. However, depending on query requirements, the composite entity's primary key may be an entirely new attribute, thus producing a table containing two candidate keys: the designated primary key and the combination of foreign keys that could have served as the primary key.

Composite entities may be named to reflect their component entities. For example, an employee may have several insurance policies (life, dental, accident, health, etc.), and each insurance policy may be held by many employees. This M:N relationship is converted to a set of two 1:M relationships, by creating a composite entity named EMP_INS. The EMP_INS entity must contain at least the primary key components of each of the two entities connected by it. How many additional attributes are kept in the composite entity depends on the end-user information requirements.
15. Given the following business rules, create the appropriate Chen and Crow’s Foot E-R diagram for each of the specified relationships:

   a. A company operates four departments.

   Although the company operates four departments now, it may later decide to combine operations or to expand them. Clearly, the company operates at least one department.

   b. Each department in part (a) employs employees.

   To exist, a department must have at least one employee. To simplify payroll procedures, an employee is likely to be assigned to just one department. How many people must be involved in a given operation before a department can be formed? The answer to that question depends on company policy, which dictates the appropriate business rule. Thus the business rule(s) will ultimately decide the cardinalities.

   c. Each of the employees in part b may or may not have one or more dependents.

   Since an employee is not required to claim a dependent, DEPENDENT is optional to EMPLOYEE. Because a company cannot permit the existence of a dependent who is not claimed by an employee, and because we have chosen to make the DEPENDENT's primary key a combination of the EMPLOYEE's primary key and the DEPENDENT's number, DEPENDENT is a weak entity.

   d. Each employee in part (c) may or may not have an employment history.

   Each employee may have an employment history. If a newly hired employee has no prior work experience, that employee will not have an employment history, so the EMP_HIST is optional to EMPLOYEE. Since an employee may have worked for more than one previous employer or may have worked in several other departments prior to the current assignment, there is a 1:M relationship between EMPLOYEE and EMP_HIST. Clearly, an employment history is associated with a specific employee and, because we have decided that the EMP_HIST' primary key will be a combination of the EMPLOYEE's primary key and the EMP_HIST's record number, the EMP_HIST is both a weak and optional entity. In short, the EMP_HIST dependencies and primary key may be:

   EMP_HIST (EMP_NUM, HIST_LINE, HIST_YEAR, HIST_DESCRIPTION, .... etc.)

   (We have bold-faced the primary key components.) Note again that the EMP_HIST is a weak entity because of the designer's decision to create its primary key in this fashion.

16. Using the E-R diagram components developed in question 15, create Chen and Crow’s Foot E-R
diagrams that includes all the components.
17. What three (often conflicting) database requirements must be addressed in database design?

Database design must reconcile the following requirements:

- Design elegance requires that the design must adhere to design rules concerning nulls, derived attributes, redundancies, relationship types, and so on.
- Information requirements are dictated by the end users.
- Operational (transaction) speed requirements are also dictated by the end users.

Clearly, an elegant database design that fails to address end user information requirements or one that forms the basis for an implementation whose use progresses at a snail's pace has little practical use.

18. Briefly, but precisely, explain the difference between single-valued attributes and simple attributes. Give an example of each.

A single-valued attribute is one that can have only one value. For example, a person has only one first name and only one social security number. A simple attribute is one that cannot be decomposed into its component pieces. For example, a person's sex is classified as either M or F and there is no reasonable way to decompose M or F. Similarly, a person's first name cannot be decomposed into meaningful components. (In contrast, if a phone number includes the area code, it can be decomposed into the area code and the phone number itself. And a person's name may be decomposed into a first name, an initial, and a last name.)

Single-valued attributes are not necessarily simple. For example, an inventory code HWPRIJ23145 may refer to a classification scheme in which HW indicates HardWare, PR indicates Printer, IJ indicates InkJet, and 23145 indicates an inventory control number. Therefore, HWPRIJ23145 may be decomposed into its component parts... even though it is single-valued. To facilitate product tracking, manufacturing serial codes must be single-valued, but they may not be simple. For instance, the product serial number TNP5S2M231109154321 might be decomposed this way:

| TN | = state = Tennessee |
| P5 | = plant number 5 |
| S2 | = shift 2 |
| M23 | = machine 23 |
| 11 | = month, i.e., November |
| 09 | = day |
| 154321 | = time on a 24-hour clock, i.e., 15:43:21, or 3:43 p.m. plus 21 seconds |
19. What are multivalued attributes, and how can they be handled within the database design?

As the name implies, multi-valued attributes may have many values. For example, a person's education may include a high school diploma, a 2-year college associate degree, a four-year college degree, a Master's degree, a Doctoral degree, and various professional certifications such as a Certified Public Accounting certificate or a Certified Data Processing Certificate.

There are basically three ways to handle multi-valued attributes, and two of those three ways are bad:

1. Each of the possible outcomes is kept as a separate attribute within the table. This solution is undesirable for several reasons. First, the table would generate many nulls for those who had minimal educational attainments. Using the preceding example, a person with only a high school diploma would generate nulls for the 2-year college associate degree, the four-year college degree, the Master's degree, the Doctoral degree, and for each of the professional certifications. In addition, how many professional certification attributes should be maintained? If you store two professional certification attributes, you will generate a null for someone with only one professional certification and you'd generate two nulls for all persons without professional certifications. And suppose you have a person with five professional certifications? Would you create additional attributes, thus creating many more nulls in the table, or would you simply ignore the additional professional certifications, thereby losing information?

2. The educational attainments may be kept as a single, variable-length string or character field. This solution is undesirable because it becomes difficult to query the table. For example, even a simple question such as "how many employees have four-year college degrees?" requires string partitioning that is time-consuming at best. Of course, if there is no need to ever group employees by education, the variable-length string might be acceptable from a design point of view. However, as database designers we know that, sooner or later, information requirements are likely to grow, so the string storage is probably a bad idea from that perspective, too.

3. Finally, the most flexible way to deal with multi-valued attributes is to create a composite entity that links employees to education. By using the composite entity, there will never be a situation in which additional attributes must be created within the EMPLOYEE table to accommodate people with multiple certifications. In short, we eliminate the generation of nulls. In addition, we gain information flexibility because we can also store the details (date earned, place earned, etc.) for each of the educational attainments. The (simplified) structures might look like those in Figure Q3.19 A-C.
Figure Q3.19 A-C  The Multivalued Attribute Solution

Database name: CH3QUESTIONS

Table name: EMPLOYEE  Table name: EMP_EDUC  Table name: EDUCATION
Figure Q3.19 D  The Relational Schema for the CH3_QUESTIONS Database

By looking at the structures shown in Figures Q3.19 A-C, we can tell that the employee named Randall earned a high school diploma in 1983, a Bachelor's degree in 1989, and a Certified Network Professional certification in 1992. If Randall were to earn a Master's degree and a Certified Data Processing certification later, we merely add another two records in the EMP_EDUC table. If additional educational attainments beyond those listed in the EDUCATION table are earned by any employee, all we need to do is add the appropriate record(s) to the EDUCATION table, then enter the employee's attainments in the EMP_EDUC table. There are no nulls, we have superb query capability, and we have flexibility. Not a bad set of design goals!

The design on which the Figure Q3.19 A-C's table structures are based is shown in Figure Q3.19E.

Figure Q3.19E The ERD for the CH3_QUESTIONS Database
The final four questions are based on the E-R diagram in Figure Q3.20.

20. Write the proper cardinalities for
   (a,b)___(0,N)___ (c,d)___(1,1)___
   (e,f)___(0,N)___
   (g,h)___(1,1)___ (i,j)___(1,1)___ (k,l)___(0,N)___ (m,n) ____0,N____ (o,p) ___1,1___

21. Write the proper connectivities for
   W___1___ W___1___ X___M___ Y___M___ Z___1___

22. What two attributes must be contained in the composite entity? Use proper terminology in your answer.

   The composite entity must at least include the primary keys of the entities it references. The combination of these attributes may be designated to be the composite entity's (composite) primary key. Each of the (composite) primary key's attributes is a foreign key that references the entities for which the composite entity serves as a bridge.

23. Describe precisely the composition of the weak entity's primary key. Use proper terminology in your answer.

   A weak entity's primary key must be a composite key that includes the primary key of the entity on which it is existence-dependent. (For example, note the detailed discussion in question 14 c and d.)

24. Convert the Chen ERD in Figure Q3.20 to a Crow’s Foot ERD.

   Please see file 3-24.gif and file 3-24.vsd on the CD and website for the Crow’s foot diagram for number 24.
Answers to Problems

The first three problems are based on the E-R model in Figure P3.1.

![E-R diagram](image)

**Figure P3.1 The ERD for Problems 1-3**

1. Use the following business rules to write all appropriate connectivities in the E-R diagram:

   a. A department employs many employees, but each employee is employed by one department.

   b. Some employees, known as "rovers," are not assigned to any department.

   c. A division operates many departments, but each department is operated by one division.

   d. An employee may be assigned to many projects, and a project may have many employees assigned to it.

   e. A project must have at least one employee assigned to it.

   f. One of the employees manages each department, and each department is managed by only one employee.

The answers to question 1 (all parts) are included in the E-R diagram that accompanies Problem 3.
g. One of the employees runs each division, and each division is run by one employee.

2. Write all the cardinalities into the model.

The answer to question 2 is included in the E-R diagram that accompanies Problem 3.

3. Modify the E-R model by splitting the M:N relationship into two 1:M relationships that are connected through a composite entity. Then rewrite the connectivities and cardinalities to match the changes you have made.

The completed Chen ERD is shown in Figure P3.3. Note that there are two relationships between DEPARTMENT and EMPLOYEE.

Figure P3.3 The Completed Chen ERD for Problems 1-3
Discussion: Note that the ERD shown in Figure P3.3a – and in the Crow’s Foot ERD shown in Figure P3.3b -- reflects several useful features that become especially important when the design is implemented. For example:

- The ASSIGN entity is shown to be optional to the PROJECT. This decision makes sense from a practical perspective, because it lets you create a new project record without having to create a new assignment record. (If a new project is started, there will not yet be any assignments.)

- The relationship expressed by “DEPARTMENT employs EMPLOYEE” is shown as mandatory on the EMPLOYEE side. This means that a DEPARTMENT must have at least one EMPLOYEE in order to have departmental status. However, DEPARTMENT is optional to EMPLOYEE, so an employee can be entered without entering a departmental FK value. If the existence of nulls is not acceptable, you can create a “No assignment” record in the DEPARTMENT table, to be referenced in the EMPLOYEE table if an employee is not assigned to a department.

- Note also the implications of the 1:1 “EMPLOYEE manages DEPARTMENT” relationship. The flip side of this relationship is that “each DEPARTMENT is managed by one EMPLOYEE”. (This latter relationship is shown as mandatory in the ERD. That is, each department must be managed by an employee!) Therefore, one of the EMPLOYEE table’s PK values must appear as the FK value in the DEPARTMENT table. (Because this is a 1:1 relationship, the index property of the EMP_NUM FK in the DEPARTMENT table must be set to “unique.”)

- Although you ought to approach a 1:1 relationship with caution – most 1:1 relationships are the result of a misidentification of attributes as entities – the 1:1 relationships reflected in the “EMPLOYEE manages DEPARTMENT” and “EMPLOYEE runs DISISION” are appropriate. These 1:1 relationships avoid the data redundancies you would encounter if you duplicated employee data – such a names, phones, and e-mail addresses – in the DIVISION and DEPARTMENT entities.
4. Convert the Chen model you have developed in problems 1-3 to a Crow’s Foot model. Include at least the minimum number of attributes required to implement the model.

If you develop the Crow’s Foot ERD shown in Figure P3.4, there are some important Crow’s Foot features to keep in mind.

- If you have multiple relationships between two entities -- such as the “EMPLOYEE manages DEPARTMENT” and “DEPARTMENT employs EMPLOYEE” relationships – you must make sure that each relationship has a designated primary entity. For example, the 1:1 relationship expressed by “EMPLOYEE manages DEPARTMENT” requires that the EMPLOYEE entity be designated as the primary (or “first”) entity. If you use Visio to create your Crow’s Foot ERDs, Figures P3.4a and 4b show how the 1:1 relationship is specified. If you use some other CASE tool, you will discover that it, too, is likely to require similar relationship specifications.

- The Crow’s Foot ERD in Figure P3.4 contains attribute information that is not available in the Chen model. In addition, the nature of the relationships (identifying or non-identifying) is immediately obvious. Note also that the ASSIGN entity was renamed ASSIGNMENT – the Crow’s Foot entity specification provided plenty of room to expand the name to its proper noun specification. (ASSIGN might be interpreted as a verb.)

Figure P3.4 The Completed Crow’s Foot ERD for Problems 1-3
5. Temporary Employment Corporation (TEC) places temporary workers in companies during peak periods. TEC’s manager gives you the following description of the business:

- TEC has a file of candidates who are willing to work.
- If the candidate has worked before, that candidate has a specific job history. (Naturally, no job history exists if the candidate has never worked. Each time the candidate worked, one additional job history record was created.)
- Each candidate has several qualifications. Each qualification may be earned by more than one candidate. (For example, it is possible for more than one candidate to have earned a BBA degree or a Microsoft Network Certification. And clearly a candidate may have earned a BBA and a Microsoft Network Certification.)
- TEC also has a list of companies that request temporaries.
- Each time a company requests a temporary employee, TEC makes an entry in the openings folder. This folder contains an opening number, company name, required qualifications, starting date, anticipated ending date, and hourly pay.
- Each opening requires only one specific or main qualification.
When a candidate matches the qualification, (s)he is given the job, and an entry is made in the Placement Record folder. This folder contains an opening number, candidate number, total hours worked, and so on. In addition, an entry is made in the job history for the candidate.

TED uses special codes to describe a candidate's qualifications for an opening. The list of codes includes:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEC-45</td>
<td>Secretarial work, at least 45 words per minute</td>
</tr>
<tr>
<td>SEC-60</td>
<td>Secretarial work, at least 60 words per minute</td>
</tr>
<tr>
<td>CLERK</td>
<td>General clerking work</td>
</tr>
<tr>
<td>PRG-VB</td>
<td>Programmer, Visual Basic</td>
</tr>
<tr>
<td>PRG-C++</td>
<td>Programmer, C++</td>
</tr>
<tr>
<td>DBA-OR</td>
<td>Database Administrator, ORACLE</td>
</tr>
<tr>
<td>DBA-DB2</td>
<td>Database Administrator, DB2</td>
</tr>
<tr>
<td>SYS-1</td>
<td>Systems Analyst, level 1</td>
</tr>
<tr>
<td>SYS-2</td>
<td>Systems Analyst, level 2</td>
</tr>
<tr>
<td>NW-NOV</td>
<td>Network administrator, Novell experience</td>
</tr>
</tbody>
</table>

TED's management wants to keep track of the following entities:

COMPANY
OPENING
QUALIFICATION
CANDIDATE
JOB_HISTORY
PLACEMENT

Given this information, do the following:

a. Draw the Crow’s Foot E-R diagram for this enterprise.

b. Identify all possible relationships.

c. Identify the connectivity for each relationship.

d. Identify the mandatory/optional dependencies for the relationships.

e. Resolve all M:N relationships.

Problems 5a-e are answered in the E-R diagram. Shown in Figure P3.5.
P3.5 The ERD for Problem 3.5

To help the students understand the E-R diagram's components better, the following discussion is likely to be useful:

- Each COMPANY may list one or more OPENINGs. Because we will maintain COMPANY data even if a company has not (yet!) hired any of TEC's candidates, OPENING is an optional entity in this relationship. OPENING is existence-dependent on COMPANY, because there cannot be an opening unless it is listed by a company. Because we have decided to use the COMPANY primary key as a component of the OPENING's primary key, we have satisfied the conditions that will allow us to classify OPENING as a weak entity.

- Each job CANDIDATE may have many job HISTORY entries. But keep in mind that a candidate may just have competed job training and, therefore, may not have had job experience (i.e., no job history) yet. In short, HISTORY is optional to CANDIDATE. On the other hand, a job candidate may have had many jobs (remember, TEC is a temp employer!) and, therefore, would have many entries in HISTORY. Finally, HISTORY is clearly existence-dependent on CANDIDATE; it is not possible to make an entry in HISTORY without having a CANDIDATE to generate that history. We will use the CANDIDATE primary key as one of the components of the HISTORY's primary key, thus allowing us to classify HISTORY as a weak entity.
Each CANDIDATE may have earned one or more QUALIFICATIONs. Although a qualification may be listed by a company, there may not be a matching candidate because it is possible that none of the candidates have this qualification. For instance, it is possible that none of the available candidates is a Pascal programmer. Therefore, CANDIDATE is optional to QUALIFICATION. However, many candidates may have a given qualification. For example, many candidates may be C++ programmers. And each qualification may be matched to many job candidates, so the relationship between CANDIDATE and QUALIFICATION is M:N. This relationship must be decomposed into two 1:M relationships with the help of a composite entity we will name CAN_QUAL.

Each job OPENING requires one QUALIFICATION, and any given qualification may fit many openings, thus producing a 1:M relationship between QUALIFICATION and OPENING. For example, a job opening for a C++ programmer requires an applicant to have the C++ programming qualification, but there may be many job openings for C++ programmers! However, a qualification does not require an opening. (After all, if there is no listing with a C++ requirement, a candidate who has the C++ qualification does not match the listing!) Therefore, OPENING is optional to QUALIFICATION. Because there cannot be a listed opening unless it also lists the required qualification for that opening, the OPENING is existence-dependent on QUALIFICATION. We will use the QUALIFICATION primary key as a component of the OPENING's (composite) primary key. Because OPENING is existence-dependent on QUALIFICATION and because it borrows the QUALIFICATION's primary key as component of its own primary key, OPENING is properly classified as a weak entity to QUALIFICATION.

A listed job opening may be filled by one or more candidates. Also, keep in mind that, during some period of time, a candidate may fill many openings. (TEC supplies temporaries, remember?) Therefore, the relationship between OPENING and CANDIDATE is M:N. We will decompose this M:N relationship into two 1:M relationships, using the composite entity named PLACEMENT as the bridge between CANDIDATE and OPENING. Because there is not necessarily an opening for each candidate, OPENING is optional to CANDIDATE. Similarly, since an OPENING may be listed even when there is no available candidate, CANDIDATE is optional to OPENING. Note the migration of the optional symbols in the E-R diagram. (You may wonder why we show OPENING as a weak entity in the E-R diagram. Actually, OPENING was declared a weak entity in our earlier discussion concerning the OPENING's relationship to COMPANY!) Also, remind the students that, although an opening may not have an available candidate, and a candidate may not fit an opening, the actual placement of a candidate clearly demonstrates that there was a match between a candidate and a listed opening!

There exists an entry in HISTORY for every PLACEMENT entry, but not all HISTORY entries will have a matching placement entry. In other words, all PLACEMENT entries will have a matching HISTORY entry, but PLACEMENT is optional to HISTORY because a position may have been filled by the candidate rather
than through the TEC services!
6. The Jonesburgh County Basketball Conference (JCBC) is an amateur basketball association. Each city in the county has one team that represents it. Each team has a maximum of 12 players and a minimum of nine players. Each team also has up to three coaches (offensive, defensive, and PT coaches.) Each team plays two games (home and visitor) against each of the other teams during the season.

Given these conditions, do the following:

a. Identify the connectivity of each relationship.

b. Identify the type of dependency that exists between CITY and TEAM.

c. Identify the cardinality between teams and players, and between teams and city.

d. Identify the dependency between coach and team, and between team and player.

e. Draw the Chen and Crow’s Foot E-R diagram to represent the JCBC database.

Problems 6a-6e are all completed in the E-R diagram shown in Figure P3.6.

Figure P3.6 The ERD for the JCBC Database
To help the students understand the E-R diagram's components better, note the following relationships:

- The main components are TEAM and GAME.
- Each team plays each other team at least twice.
- To play a game, two teams are necessary: the home team and the visitor team.
- Each team plays once as the home team and once as the visitor team.

Given these relationships, it becomes clear that TEAM participates in a recursive M:N relationship with GAME. The relationship between TEAM and GAME becomes clearer if we list some attributes for each of these entities:

<table>
<thead>
<tr>
<th>GAME entity</th>
<th>TEAM entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAME_NUM</td>
<td>TEAM_NUM</td>
</tr>
<tr>
<td>GAME_DATE</td>
<td>TEAM_NAME</td>
</tr>
<tr>
<td>GAME_HOME_TEAM</td>
<td>TEAM_CITY</td>
</tr>
<tr>
<td>GAME_VISIT_TEAM</td>
<td></td>
</tr>
<tr>
<td>GAME_HOME_POINTS</td>
<td>Note: TEAM_NUM appears at least twice in a GAME:</td>
</tr>
<tr>
<td></td>
<td>once as GAME_HOME_TEAM and once as GAME_VISIT_TEAM.</td>
</tr>
</tbody>
</table>

7. Automata Inc. produces specialty vehicles by contract. The company operates several departments, each one of which builds a particular vehicle, such as a limousine, a truck, a van, or an RV.

When a new vehicle is built, the department places an order with the Purchasing Department to request specific components. Automata's Purchasing Department is interested in creating a database to keep track of orders and to accelerate the process of delivering materials.

The order received by the Purchasing Department can contain several different items. An inventory is maintained so that the most frequently requested items are delivered almost immediately. When an order comes in, it is checked to determine whether the requested item(s) is (are) in inventory. If an item is not in inventory, it must be ordered from a supplier. Each item may have several suppliers.

Given this functional description of the processes encountered at Automata's Purchasing Department, do the following:

a. Identify all the main entities.
Note: Problems (a) through (c) are answered in the E-R diagram following answer (c).

b. Identify all the relations and connectivities among entities.

c. Identify the type of existence dependency in all relations.

![ERD Diagram for Problem 7](image)

Figure P3.7 The ERD for Problem 7

d. Give some examples of the types of reports that can be obtained from the database.

With this database the following reports can be obtained, among others:

- report of suppliers and items supplied by each.
- number of orders placed by each department, including the requested items.
- an updated inventory report showing stock quantities, components that need to be reordered, etc.

8. Create an ERD based on the Crow’s Foot model, using the following requirements.

- An INVOICE is written by a SALESREP. Each sales representative can write many invoices, but each invoice is written by a single sales representative.

- The INVOICE is written for a single CUSTOMER. However, each customer can have many invoices.

- An INVOICE may include many detail lines (LINE) which describe the
products bought by the customer.

- The product information is stored in a PRODUCT entity.
- The product's vendor information is found in a VENDOR entity.
Figure P3.8A  The Revised ERD for Problem 8

9. Given the following brief summary of business rules for the ROBCOR catering service, and using the Crow’s Foot E-R methodology, draw the fully-labeled ERD. Make sure to include all appropriate entities, relationships, connectivities, and cardinalities.

Each dinner is based on a single entree, but each entree can be served at many dinners. A guest can attend many dinners, and each dinner can be attended by many guests. Each dinner invitation can be mailed to many guests, and each guest can receive many invitations.
10. Using the Crow’s Foot methodology, create an ERD that can be implemented for a medical clinic, using at least the following business rules:

- A patient can make many appointments with one or more doctors in the clinic, and a doctor can accept appointments with many patients. However, each appointment is made with only one doctor, and each appointment references a single patient.

- Emergency cases do not require an appointment. However, an emergency is entered into the appointment book as "unscheduled" for appointment management purposes.

- If kept, an appointment yields a visit with the doctor specified in the appointment. The visit yields a diagnosis and, when appropriate, treatment.

- Each visit updates the patient's records to provide a medical history.

- Each patient visit creates a bill. Each patient visit is billed by one doctor, and each doctor can bill many patients.

- Each bill must be paid. However, a bill may be paid off in many installments, and a payment may cover more than one bill.

- A patient may pay the bill directly, or the bill may be the basis for a claim submitted to an insurance company.
If the bill is paid by an insurance company, the deductible is submitted to the patient for payment.

Figure P3.10 The ERD for Problem 10
11. Tiny College is so pleased with your design and implementation of its student registration/tracking system that it wants you to expand the design to include its motor pool. A brief description of operations follows:

Faculty members may use the Tiny College-owned vehicles for officially-sanctioned travel. For example, its vehicles may be used by faculty members to travel to off-campus learning centers, to travel to locations at which research papers are presented, to transport students to officially sanctioned locations, and to travel for public service purposes. The vehicles used for such purposes are managed by Tiny College's TFBS (Travel Far But Slow) Center.

Using reservation forms, each department may reserve vehicles for its faculty, who are responsible for filling out the appropriate trip completion form at the end of each trip. The reservation form includes the expected departure date, vehicle type required, destination, and the authorized faculty member. When the faculty member arrives to pick up the vehicle, (s)he must sign a check-out form to log the vehicle out and to pick up a trip completion form. (The TFBS employee who releases the vehicle for use also signs the check-out form.) The faculty member's trip completion form includes the faculty member's identification code, the vehicle's identification, the odometer readings at the start and end of the trip, maintenance complaints, if any, gallons of fuel purchased, if any, and the Learnwell College credit card used to pay for such fuel. If fuel has been purchased, the credit card receipt must be stapled to the trip completion form. Upon receipt of the Faculty Trip Completion form, the faculty member's department is billed at a mileage rate based on the vehicle type (sedan, station wagon, panel truck, minivan, minibus) used. HINT: Do NOT use more entities than are necessary. Remember the difference between attributes and entities!

All vehicle maintenance is performed by TFBS. Each time a vehicle requires maintenance, a maintenance log entry is completed on a pre-numbered maintenance log form. The maintenance log form includes the vehicle identification, a brief description of the type of maintenance required, the initial log entry date, the date on which the maintenance was completed, and the identification of the mechanic who released the vehicle back into service. (Only mechanics who have an inspection authorization may release the vehicle back into service.)

As soon as the log form has been initiated, the log form's number is transferred to a maintenance detail form; the log form's number is also forwarded to the parts department manager, who fills out a parts usage form on which the maintenance log number is recorded. The maintenance detail form contains separate lines for each maintenance item performed, the parts used, and the identification of the mechanic who performed the maintenance item. When all the maintenance items have been completed, the maintenance detail form is stapled to the maintenance log form, the maintenance log form's completion date is filled out, and the mechanic who releases the vehicle back to service signs the
form. The stapled forms are then filed, to be used later as the source for various maintenance reports.

TBFS maintains a parts inventory, including oil, oil filters, air filters, belts of various types, and so on. The parts inventory is monitored daily to monitor parts usage and to re-order parts that reach the "minimum quantity on hand" level. To track parts usage, the parts manager requires each mechanic to sign out the parts that are used to perform each vehicle's maintenance; the parts manager records the maintenance log number under which the part is used.

Each month, TFBS issues a set of reports. These reports include the mileage driven by vehicle, by department, and by faculty members within the department. In addition, various "revenue" reports are generated by vehicle and department. A detailed parts usage report is also filed each month. Finally, a vehicle maintenance summary is created each month.

Given this brief summary of operations, draw the appropriate (and fully-labeled!) E-R diagram. Use the Chen methodology to indicate entities, relationships, connectivities, and cardinalities.

Figure P3.11 The ERD for Problem 11
Figure P3.11's E-R diagram has been rendered at the implementation level. That is, we have converted all composite entity relationships by naming them, thus eliminating the need to show the diamonds within the boxes. The following conditions are reflected within the E-R diagram:

- Because a vehicle can require maintenance many times and each maintenance procedure requires a new log entry, the relationship between VEHICLE and LOG is 1:M. Because even a new vehicle is checked initially through the maintenance department, the relationship is mandatory.

- The MAINTENANCE entity is shown as weak to LOG, because the maintenance detail is defined partially by the log number and, quite clearly, a maintenance detail form is existence-dependent on the log entry.

- Some maintenance does not require parts. For example, the adjustment of a fuel injector jet only requires a mechanic's time. Therefore, PART is optional to MAINTENANCE in this relationship. On the other hand, a given vehicle may use many parts during a maintenance operation.

- If a part is required, it must be signed out. Any part can be signed out many times. For example, if the parts inventory includes 25 hose clamps, a hose clamp can be signed out 25 times. (Not the same one, of course.... but each time a hose clamp is used, that hose clamp part number shows up in the SIGN_OUT!)

- Each maintenance line must be signed off by a mechanic and, if a mechanic does any maintenance work, that mechanic is required to sign off on that work. If a mechanic performs many tasks on a given vehicle during its maintenance, that mechanic signs off many times, once for each completed task.

- Only mechanics who have an inspection authorization can sign off the LOG, so LOG is optional to MECHANIC in the MECHANIC creates LOG (entry). But a mechanic with an inspection authorization can sign off many logs.

- Not all departments make reservations for the use of Tiny College vehicles, so RESERVATION is optional to DEPARTMENT. On the other hand, if a reservation is made, it must have been made by a department, so DEPARTMENT is mandatory to RESERVATION.

- Faculty members may take many trips during some period of time, thus generating many check-outs. However, not all faculty members use the Tiny College vehicles, so CHECK_OUT is optional to FACULTY.

- Each check-out generates a charge and each charge is related to one check-out.
Each charge is determined by the number of miles driven (recorded in CHECK_OUT) and the charge per mile recorded by vehicle TYPE.

The students may find it helpful if the instructor also shows the composite relationships at the (more conceptual) composite level. It may also be useful to show several different types of composite entities that perform the same kinds of functions. For example, note the similarities between the E-R segments shown in Figure P3.10A.

**Figure P3.11A The ERD Segments at the Conceptual Level**

It may also be useful to discuss a scenario in which the RESERVATION, CHECK_OUT, and CHARGE attributes are kept in a single table, thus simplifying the design by eliminating the 1:1 relationships between RESERVATION, CHECK_OUT and CHARGE. The combined entity, named RES_CHKOUT, would then yield a table with these attributes:

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Sample values</th>
</tr>
</thead>
<tbody>
<tr>
<td>RES_NUM</td>
<td>21334</td>
</tr>
<tr>
<td>RES_DATE</td>
<td>10/15/96</td>
</tr>
<tr>
<td>EMP_NUM</td>
<td>4355</td>
</tr>
<tr>
<td>DEPT_CODE</td>
<td>CIS</td>
</tr>
<tr>
<td>TYPE_CODE</td>
<td>S4</td>
</tr>
<tr>
<td>RES_PICK_UP</td>
<td>11/7/96</td>
</tr>
<tr>
<td>VEH_NUM</td>
<td>FHG-1234</td>
</tr>
<tr>
<td>FAC_NUM</td>
<td>2514</td>
</tr>
<tr>
<td>RES_RETURN</td>
<td>11/8/96</td>
</tr>
<tr>
<td>RES_MILES</td>
<td>285.4</td>
</tr>
<tr>
<td>RES_COMPLAINTS</td>
<td>Dashboard rattles at low rpm, rough idling</td>
</tr>
<tr>
<td>RES_COST</td>
<td>$8.50</td>
</tr>
<tr>
<td>RES_REF</td>
<td>VISA</td>
</tr>
<tr>
<td>RES_BILL</td>
<td>$85.62</td>
</tr>
</tbody>
</table>
The problem with the preceding solution is that many of the attributes are set to null until the vehicle is returned. In fact, the vehicle and faculty numbers are set to null until the vehicle is picked up.

12. Given the following information, produce an ERD – based on the Crow’s Foot model – that can be implemented. Make sure to include all appropriate entities, relationships, connectivities and cardinalities.

EverFail company is in the quick oil & lube business. Although customers bring in their cars for what is described as quick oil changes, EverFail also replaces windshield wipers, oil filters, and air filters, subject to customer approval. The invoice contains the charges for the oil used, all parts used, and a standard labor charge. When the invoice is presented, customers pay cash, use a credit card, or write a check. EverFail does not extend credit. EverFail's database is to be designed to keep track of all components in all transactions.

Given the high parts usage of the business operations, EverFail must maintain careful control of its parts (oil, wipers, oil filters, air filters) inventory. Therefore, if parts reach their minimum on hand quantity, the part in question must be reordered from an appropriate vendor. EverFail maintains a vendor list which contains both the vendors actually used as well as potential vendors.

Periodically, EverFail mails updates to customers, based on the date of the car's service. EverFail also tracks each customer's car mileage.

The solution to this problem is found in Figure P3.12.
P3.12 The ERD for Problem 12
13. Use the following descriptions of the operations of RC_Models Company to complete this exercise.

A.

- A customer may generate many invoices
- Each invoice is generated by only one customer
- Some customers have not (yet) generated an invoice
- Some customers come from FineScale Modeler’s magazine (? I only put this in here because I suspect that the company will want to know how many people from FineScale Modeler’s magazine became customers)
- A customer has one address
- A customer can ship to many addresses
- A billing address and shipping address can be different
- A product belongs to at least one product category (?? The description does not indicate more than two categories, model and decals, but a better design would include the possibility for multiple overlapping categories – ie. Not only model and decals but WWII, SciFi, Railroad, etc.)
- A product is made by at least one manufacture
- A product can be made by many manufactures
- A product is a product at a particular scale (the description claims that a product has many scales, but things like units in stock, reorder #, manufacturer part number etc work better if a product has a single scale and then are grouped by product group – I also tossed around the idea of having something called an ‘item’ that was a product at a particular scale but it degenerated into a product that is part of a product group).
- A product has n units in stock
- A product has a retail (customer) price
- A product has a low water mark (reorder product when stock drops below this value)
- A product has a last reorder date
- A product has a last sold date (last time this product was sold to a customer)
- An invoice has 1 or more products
- An invoice has a shipping charge
- An invoice has sales tax
- An invoice must be in one and only one state (‘order created’, ‘back ordered’, ‘checking credit’, ‘waiting for packing’, ‘billed’, ‘waiting for shipping’, and ‘shipped’)
- An invoice has a timestamp for each state change
- All items in a ‘waiting for packing’ order must be in stock
- An invoice has a payment voucher for the correct amount
- An invoice has a shipping address
- A payment voucher has a card number, billing address, expiration date, holder name, …? (NOTE: a better design would have a payment voucher abstraction so
the database could handle ‘store credit’ for returned items, gift cards, and coupons)

- A product request has a customer
- A product request has a date
- A product request has one or more products that information was requested about
- A manufacturer has an address
- A manufacturer has an order web site
- A manufacturer has a minimum order size requirement by product
- A reorder slip has an ordered timestamp and an order received timestamp
- A reorder slip has a list of products, quantities, and price
- A manufacture/product pair has a whole sale (RC_Models) price

Simplifications of the model

- Remove source code from customer
- Remove product category and category map tables and replace with a category type field in the products table
- Remove invoice state table and replace with code or text in invoice table
B. Please see the additional files named 3-13.gif, 3-13.vsd, and 3-131_raster.gif located on the CD and website.

14. Use the following description of the operations of the RC_Charter2 Company to complete this exercise.

A. 
- A customer may request many charter trips
- A charter trip is requested by only one customer
- Some customers have not (yet) requested a charter trip
- An employee may be assigned to many charter trips
- Each charter trip may have many employees assigned to it
- Each charter trip has one and only one plane assigned to it
- Each charter trip may have zero or more passengers
- Each charter trip may have zero or more items of cargo
- Each charter trip may have zero or more associated crew expenses
- Each charter trip has two or more legs
- Each charter trip may have zero or more customer special charges
- Each customer has a line or credit for zero dollars (US) or more
- Each customer has used zero dollars (US) or more of their credit on prior trips
- Each customer may have made zero or more payments to their account
- Each customer may have made one or more special requests resulting in one or more special charges on one or more trips
- A special charge applies to one and only one customer and trip
- Each leg of a trip may have incurred a wait time
- Each leg of a trip consumes fuel
- Each leg of a trip has a departure and arrival time
- Each leg of a trip belongs to one and only one trip
- Each plane can be assigned to zero or more trips (but NOT at the same time!)
- Each plane is of one and only one type of plane
- Their does not have to be a plane of a particular plane type in the fleet
- There can be more than one plane of a particular plane type in the fleet
- Each type of plane can hold up to n passengers
- Each type of plane can hold up to n tons of cargo
- Each type of plane can hold up to m cubic feet of cargo
- Each type of plane can fly up to n miles on one leg of a trip
- Each type of plane requires a crew with a specific set of functional requirements (functional requirements are things like ‘load master’, ‘instrument only pilot’, …)
- Each type of requirements can be specified zero or more times for each type of plane
- A single crew member can fulfill zero or more functions
A job function has a per mile rate and an hourly rate (the rate charged is determined if the function is required by law or by the customer).

A customer may require zero or more additional job functions (possibly new crew members, possibly upgraded crew members).

Each job function has one or more tests that are required to be taken and passed at regular and irregular intervals to ensure regulatory compliance.

A single test may be taken by the same employee zero or more times.

A single employee can take zero or more different tests.

A single employee can take the same test zero or more times.

A log of all tests taken by the employee while at the company and just prior to the company will be kept in the database.

A single employee can fill out zero or one expense report per trip.

A single employee can fill out zero or more expense reports over time.

An expense report can contain one or more expense items.

An expense item can only be in one and only one expense report.

Each expense item has a cost and a date.

Each expense item is associated with one and only one expense type (food, hotel, local transportation, …).

Each expense type can be associated with zero or more expense items.

NOTES:
On the design I ignored the certificates and ratings and instead reduced the ability to perform a function (like flying in bad weather conditions) to the tests that one must pass to get the certificates. The assumption is that the passing of the tests confers the certificate (and keeping the tests current).

Tests include actually written, oral, flight, and medical tests as well as required re-certification training/classes (which I suspect involve a test at the end).

I did not hook up the address table everywhere that it is necessary because it would cross a lot of lines and make the diagram even harder to understand.

I should have probably come up with a person table that contains first_name, last_name, middle_initial, phone_number, and home_address and then had the customer, passenger, and employee tables reference that table. Once again doing so makes the diagram more confusing. (But it would capture the fact that a customer or employee can be a passenger.)

It probably makes sense to have a total_cost column in the Trips table, however, this value can be calculated from the values in the TripLegs, PlaneTypes, Function, ExpenseReport, and SpecialCharges tables.

A lot of the verification of this database will have to be done using application or database code. (for example verifying that a plane is not scheduled for a trip with a leg that is longer than it’s flying range).

Can a pilot also be the flight attendant? Can the co-pilot be the flight attendant? The example does claim that a pilot can also be the load master so it is possible. I’ve fixed the pilot/co-pilot issue (as well as the need for more than one flight attendant) by having a
‘number’ column in the CrewRequirements table to indicate if you need two separate employees with the same skills.
I’m not sure how you would calculate the rate charged a customer if they required a pilot upgrade (the law requires one rate, the hourly rate is normally used however in the case of a customer specified function).