3.1 Discovery

Discovery

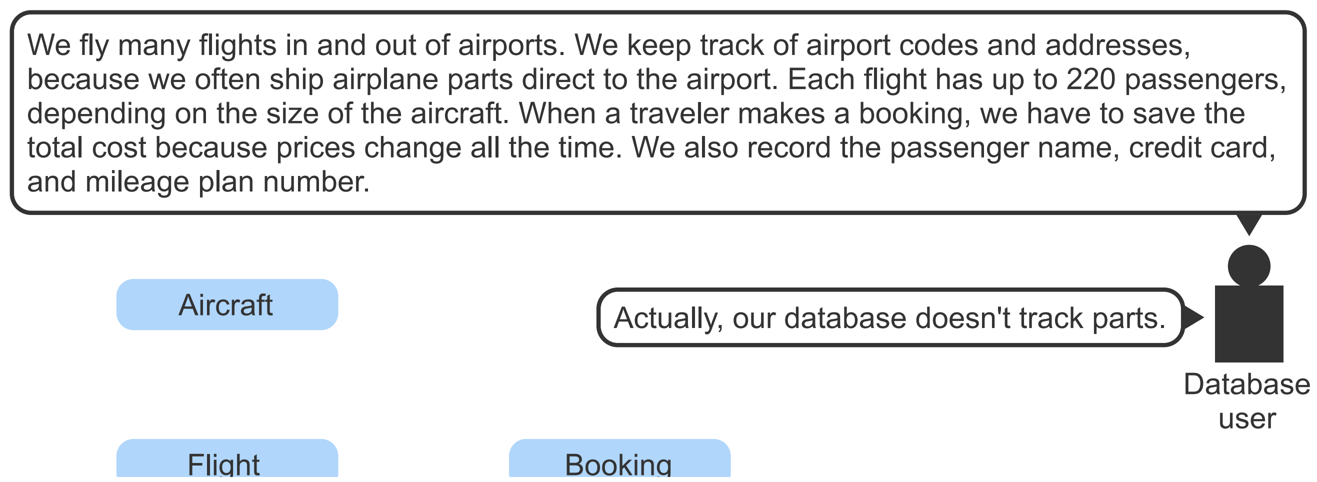
Database requirements are determined by interviewing the database users and managers. Users and managers are usually familiar with requirements from an old database, or perhaps a manual process with paper records. When users are difficult to reach, a database designer may communicate with surrogates. Ex: A sales representative might communicate on behalf of prospective customers.

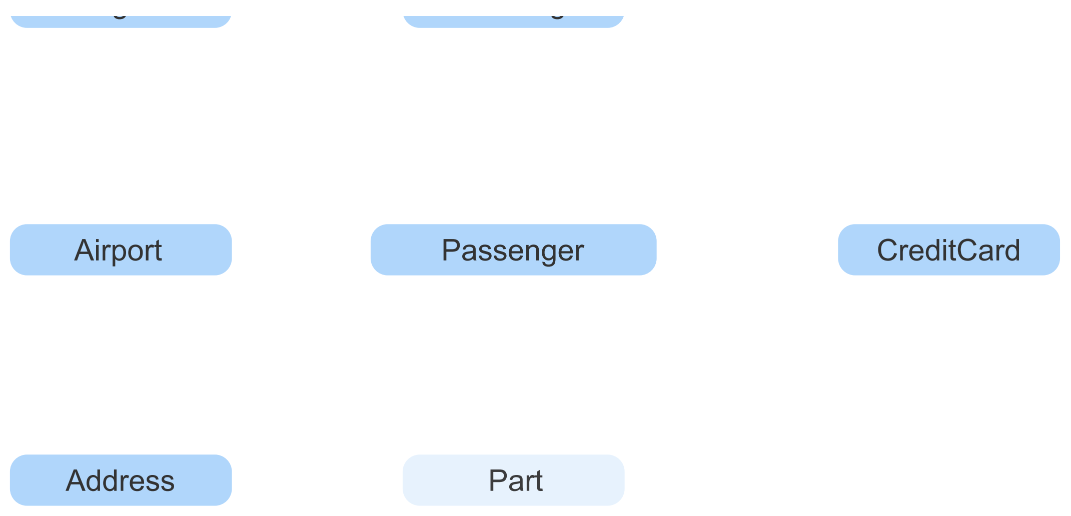
Entities, relationships, and attributes surface as nouns and verbs in an interview:

- Entities are usually nouns, but not all nouns are entities. Designers should ignore nouns that denote specific data or are not relevant to the database.
- Relationships are usually verbs. Designers should ignore statements that are not about entities, not relevant to the database, or redundant to other relationships. Designers should look for relationships that are not explicitly stated, since users may overlook important information.
- Attributes are usually nouns that denote specific data, such as names, dates, quantities, and monetary values.

In addition to interviews, written documents are a good source of data requirements. Ex: The user manual for an older version of the database is a good source of requirements.

PARTICIPATION ACTIVITY 3.1.1: Discover entities.


We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number.
Aircraft
Actually, our database doesn't track parts.
Database user
Flight
Booking



Animation content:

Step 1: In an interview with a database user, flight, airport, aircraft, passenger, and booking are entities. There is a database user who states We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number. From this statement keyword Aircraft Flight Airport Passenger and Booking are underlined and entities are created with these keywords.

Step 2: Airplane is another word for aircraft. Traveler is another word for passenger. Keywords airplane and traveler get underlined in the statement by the database user. Entities Aircraft and Passenger are boxed.

Step 3: Usually, address and credit card have many details and are entities. Keywords Address and Credit Card are underlined in the statement by the database user and entities are created for these keywords.

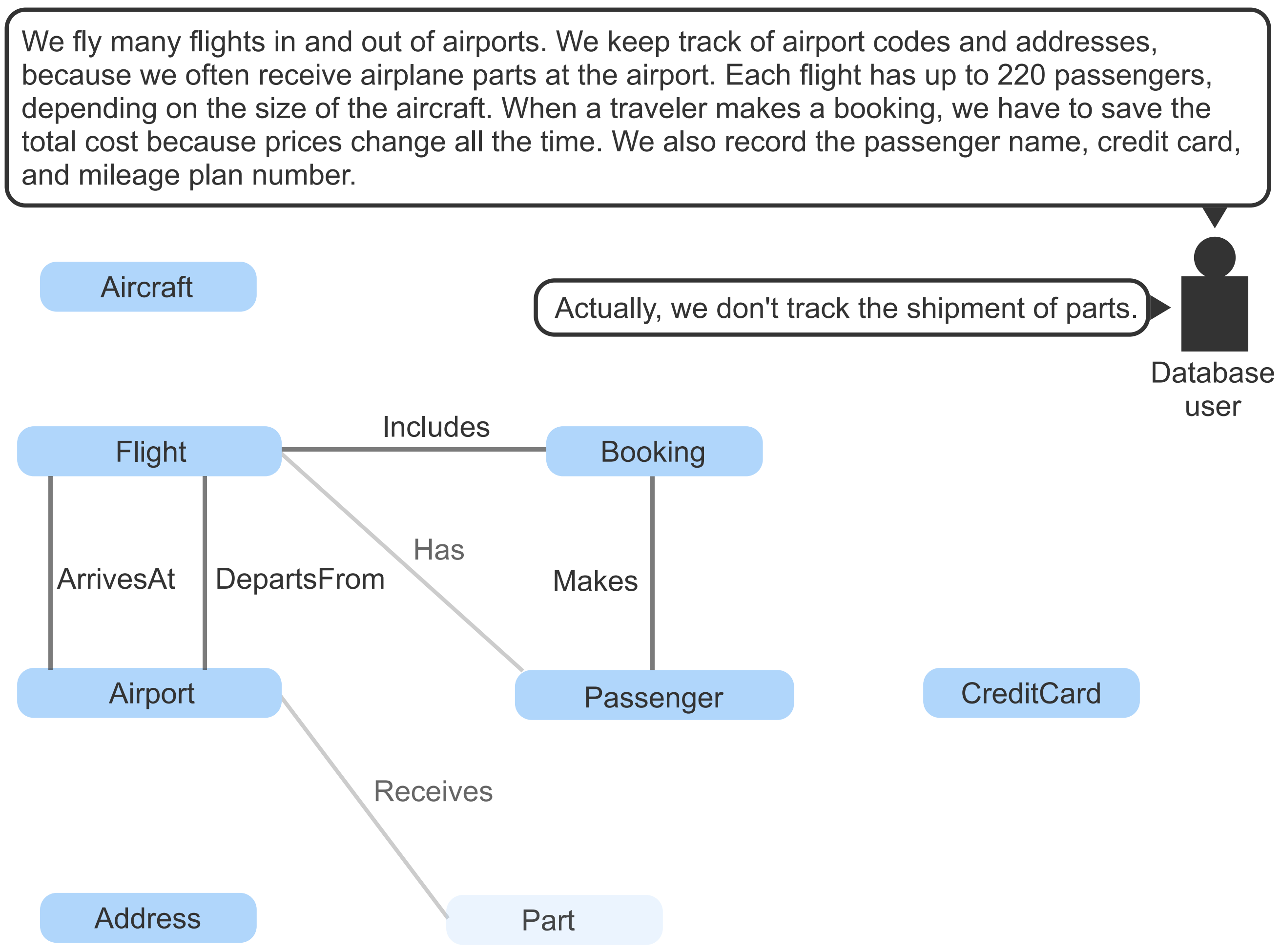
Step 4: Further interviews determine the database does not track parts. Part is not an entity. Keyword parts is underlined in the statement by the database user and an entity is created for this keyword but is then faded away.

Animation captions:

1. In an interview with a database user, flight, airport, aircraft, passenger, and booking are entities.
2. Airplane is another word for aircraft. Traveler is another word for passenger.

3. Usually, address and credit card have many details and are entities.
4. Further interviews determine the database does not track parts. Part is not an entity.

PARTICIPATION ACTIVITY | 3.1.2: Discover relationships.


We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often receive airplane parts at the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number.
Aircraft
Actually, we don't track the shipment of parts.
Database user
Flight
Includes
Booking
ArrivesAt
DepartsFrom
Has
Makes
Airport
Passenger
CreditCard
Receives
Address
Part

Animation content:

Step 1: Flight-ArrivesAt-Airport, Flight-DepartsFrom-Airport, and Passenger-Makes-Booking are relationships. There is a database user who states We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number. There are also eight entities named Aircraft, Flight, Airport, Address, Booking, Passenger, Part and Credit Card. Keywords fly, in, out, and makes are underlined in the statement by the database user. Two lines named Arrives At and Departs From connect entities Flight and Airport. A line named Makes connects entities Booking and Passenger.

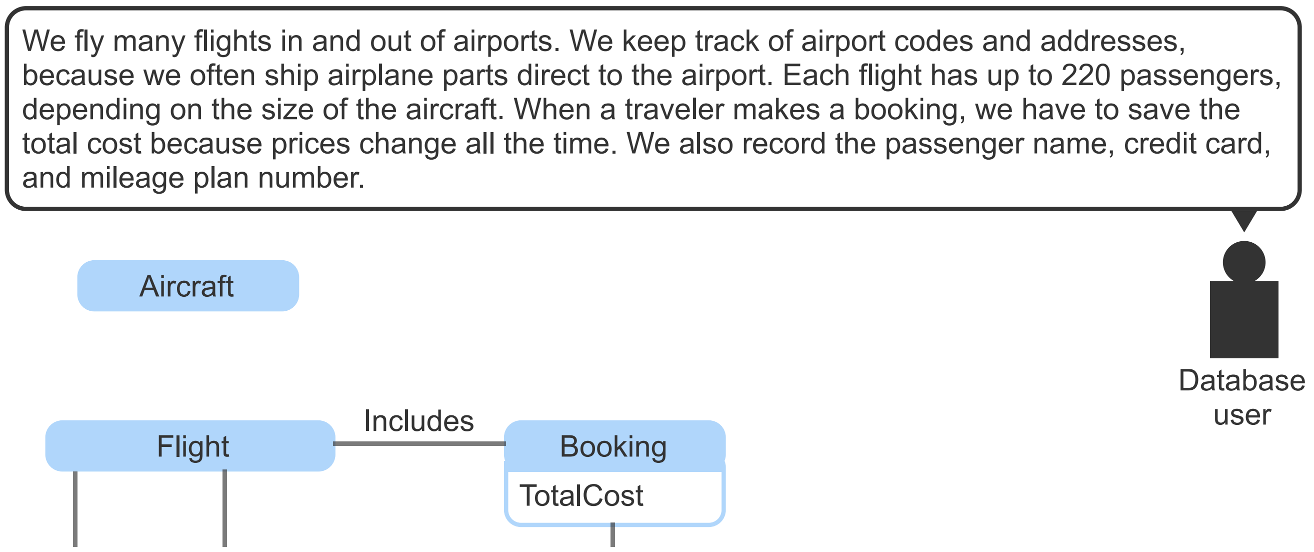
Step 2: Passengers on flights can be determined from Flight-Includes-Booking and Passenger-Makes-Booking. So Flight-Has-Passenger is unnecessary. Keyword Has is underlined in the statement by the database user. A line named Has connects entities Flight and Passenger. A line named Includes connects entities Flight and Booking and causes the line Has to fade.

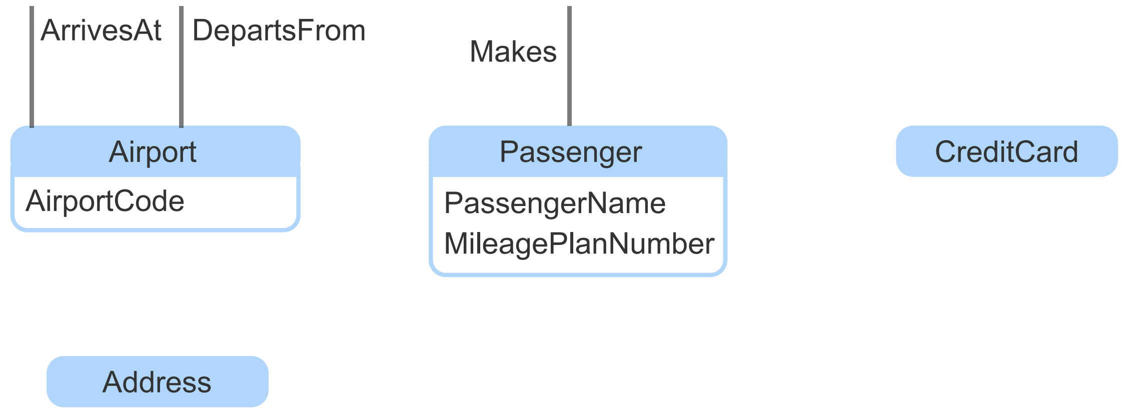
Step 3: Further interviews determine the database does not track parts. Airport-Receives-Part is not a relationship. Keyword ship is underlined in the statement by the database user. A line named Shipped To connects entities Airport and Part and then the line Shipped To and entity Part fades.

Animation captions:

1. Flight-ArrivesAt-Airport, Flight-DepartsFrom-Airport, and Passenger-Makes-Booking are relationships.
2. Passengers on flights can be determined from Flight-Includes-Booking and Passenger-Makes-Booking. So Flight-Has-Passenger is unnecessary.
3. Further interviews determine the database does not track parts. Airport-Receives-Part is not a relationship.

PARTICIPATION ACTIVITY | 3.1.3: Discover attributes.


We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number.



Animation content:

Step 1: Airport code, total cost, and name are attributes of Airport, Booking, and Passenger. There is a database user who states We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number. There are also seven entities named Aircraft, Flight, Airport, Address, Booking, Passenger, and Credit Card. Two lines named Arrives At and Departs From connect entities Flight and Airport. A line named Makes connects entities Booking and Passenger. A line named Includes connects entities Flight and Booking. Keywords airport codes are highlighted and Airport Code are added as an attribute to entity Airport. Keywords total cost is highlighted and Total Cost is added as an attribute to entity Booking. Keywords passenger name is highlighted and Passenger Name is added as an attribute to entity Passenger.

Step 2: Mileage plan number is an attribute of Passenger or, if additional mileage plan information is tracked, a separate entity. Keywords mileage plan number are highlighted and Mileage Plan Number are added as an attribute to entity Passenger.

Animation captions:

1. Airport code, total cost, and name are attributes of Airport, Booking, and Passenger.
2. Mileage plan number is an attribute of Passenger or, if additional mileage plan information is tracked, a separate entity.

PARTICIPATION ACTIVITY | 3.1.4: Discovery.

Refer to the following interview:

Our department tracks student information. We have a record of every course taken by students and the professor who taught the course. We also keep track of the name and number of credits for each course. For professors, we always store the name and current title, such as "Assistant Professor" or "Visiting Lecturer".

1) Which noun is an entity?

- ○ information
- ○ name
- ○ professor

2) Which verb is a relationship?

- ○ taken
- ○ track
- ○ have

3) Which noun is an attribute?

- ○ record
- ○ Assistant Professor
- ○ title
- ○ course

Names

Entity names are a singular noun. Ex: Employee rather than Employees. The best names are commonly used and easily understood by database users.

Relationship names have the form `Entity-Verb-Entity`, such as Division-Contains-Department. When the related entities are obvious, in ER diagrams or informal conversation, `Verb` is sufficient and entity names can be omitted. The verb should be active rather than passive. Ex: Manages rather than IsManagedBy. Occasionally, the same verb relates different entity pairs. Ex: Order-Contains-LineItem and Division-Contains-Department.

Attribute names have the form `EntityQualifierType`, such as EmployeeFirstName:

- `Entity` is the name of the entity that the attribute describes. When the entity is obvious, in ER diagrams or informal conversation, `QualifierType` is sufficient and the entity name can be omitted.
- `Qualifier` describes the meaning of the attribute. Ex: First, Last, and Alternate. Sometimes a qualifier is unnecessary and can be omitted. Ex: StudentNumber.

- `Type` is chosen from a list of standard attribute types such as Name, Number, and Count. Attribute types are not identical to SQL data types. Ex: "Amount" might be an attribute type representing monetary values, implemented as the MONEY data type in SQL. "Count" might be an attribute type representing quantity, implemented as NUMBER in SQL.

Standard attribute types are documented in the glossary and applied uniformly to all attribute names.

Table 3.1.1: Example names.

	Formal name	Informal name
Entity	Vehicle	Vehicle
Relationship	Vehicle-BelongsTo-Person	BelongsTo
Attribute	VehicleLicenseNumber	LicenseNumber

PARTICIPATION ACTIVITY 3.1.5: Naming conventions.

1) "Students" is a good entity name.
   - ○ True
   - ○ False

2) "License" is a good attribute name.
   - ○ True
   - ○ False

3) "Employee-Manages-Employee" is a good relationship name.
   - ○ True
   - ○ False

4) "People" is a good entity name.
   - ○ True
   - ○ False

5) "PassengerMileagePlanCode" is a good attribute name.

- True
- False

6) "Department-IsManagedBy-Employee" is a good relationship name.

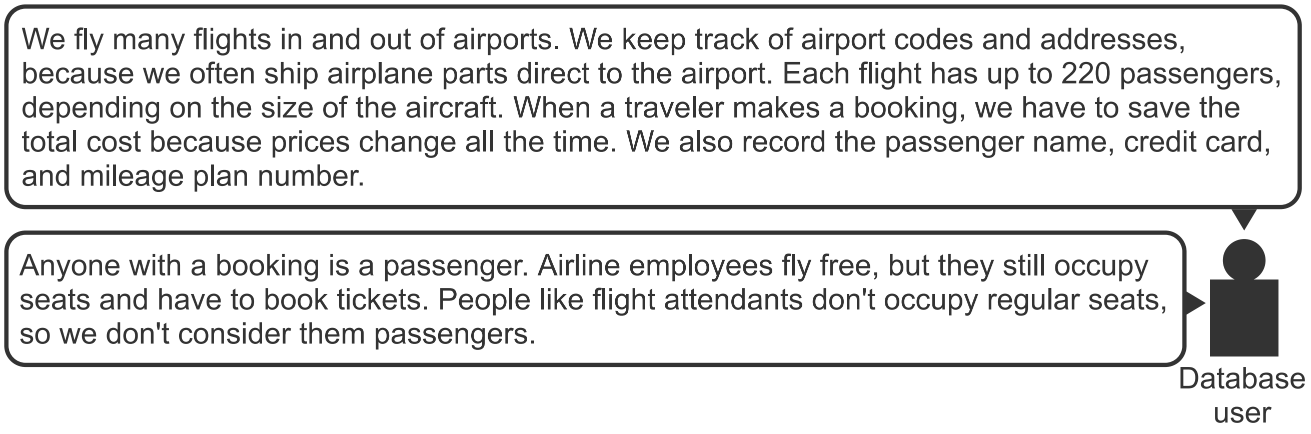
- True
- False

Synonyms and descriptions

Often, entity, relationship, and attribute names have synonyms. Ex: Representative may be a synonym for SalesAgent. Synonyms are common in informal communications. To avoid confusion, one official name is selected for each entity, relationship, and attribute. Other names are documented in the glossary as synonyms.

The glossary also contains complete descriptions of entities, relationships, and attributes. The description states the meaning of each entity, relationship, or attribute in complete sentences. The description begins with the name and includes examples and counterexamples to illustrate usage.

PARTICIPATION ACTIVITY 3.1.6: Synonyms and descriptions.


We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number.
Anyone with a booking is a passenger. Airline employees fly free, but they still occupy seats and have to book tickets. People like flight attendants don't occupy regular seats, so we don't consider them passengers.
Database user

```
Glossary

Entity Name: Passenger

Synonyms: Traveler, Customer
```

```
Description: A passenger is any person who occupies a seat on a flight.
Passengers include both paid and unpaid seat occupants, but excludes
flight officers, flight attendants, and in-cabin pets.
```

Animation content:

Step 1: Traveler is a synonym of passenger. There is a database user who states We fly many flights in and out of airports. We keep track of airport codes and addresses, because we often ship airplane parts direct to the airport. Each flight has up to 220 passengers, depending on the size of the aircraft. When a traveler makes a booking, we have to save the total cost because prices change all the time. We also record the passenger name, credit card, and mileage plan number. Keyword traveler is underlined and text with Title Glossary appears. Underneath the title is a line that separates it from the two lines Entity Name colon Passenger and Synonyms colon Traveler comma Customer.

Step 2: An entity description begins with a name and definition, followed by examples and counterexamples. The database user also states Anyone with a booking is a passenger. Airline employees fly free, but they still occupy seats and have to book tickets. People like flight attendants don't occupy regular seats, so we don't consider them passengers. This causes text to appear underneath the existing lines of text and states Description colon A passenger is any person who occupies a seat on a flight. Passengers include both paid and unpaid seat occupants, but excludes flight officers, flight attendants, and in-cabin pets.

Animation captions:

1. Traveler is a synonym of passenger.
2. An entity description begins with a name and definition, followed by examples and counterexamples.

PARTICIPATION ACTIVITY 3.1.7: Synonyms and descriptions.

1) Must entity, relationship, and attribute synonyms follow naming conventions?

- ○ Yes
- ○ Only attribute synonyms must follow naming conventions.
- ○ No

2) What is wrong with the following entity description?

The difference between a course and a class is that a course refers to the catalog description, while a class is an individual offering of a course in a specific term.

- ○ Description does not begin with entity name.
- ○ Description does not include counterexamples.
- ○ Description does not use complete sentences.

3) What is wrong with the following relationship description?

"Student-Takes-Course" describes all course instances taken by a student instance.

- ○ Description should begin with an entity name.
- ○ Description does not use complete sentences.
- ○ Description does not include examples and counterexamples.

Database design

The first step of the analysis phase is discovery of entities, relationships, and attributes in interviews and document review. As discovery proceeds, the designer draws an ER diagram, determines standard attribute types, and documents names, synonyms, and descriptions in the glossary.

Although the step numbers suggest a sequence, database designers commonly move back and forth between steps. As names, synonyms, and descriptions are documented, additional entities, relationships, and attributes are discovered. The ER diagram and glossary are usually developed in parallel.

Table 3.1.2: Discover entities, relationships, and attributes.

Step	Activity
1A	Identify entities, relationships, and attributes in interviews.
1B	Draw ER diagram.
1C	List standard attribute types in glossary.
1D	Document names, synonyms, and descriptions in glossary.

PARTICIPATION ACTIVITY 3.1.8: Database design.

1) Discovery is a step in which phase?
- ○ Analysis
- ○ Logical design
- ○ Database design

2) Identification of entities, relationships, and attributes precedes documentation.
- ○ Always
- ○ Usually
- ○ Never

3) Standard attribute types are determined after the ER diagram is drawn.
- ○ Always
- ○ Usually
- ○ Never

CHALLENGE ACTIVITY | 3.1.1: Discovery.

544874.3500394.qx3zqy7

Start

Our state's Department of Education maintains a database of colleges in the state. Each c
name, a code, and the count of students currently enrolled. Each college may have more tl
campus. Each campus has a name that is unique within the college and has an address. N
campus of a college may be located in the same city. Cities are identified by name. A city r
campuses of more than one college.

What are the entities?

- [ ] City
- [ ] Campus
- [ ] Student
- [ ] College
- [ ] Name
- [ ] State

1 2 3

Check Next

3.2 Entities, relationships, and attributes

The entity-relationship model

Database design begins with verbal or written requirements for the database. Requirements are formalized as an entity-relationship model and then implemented in SQL.

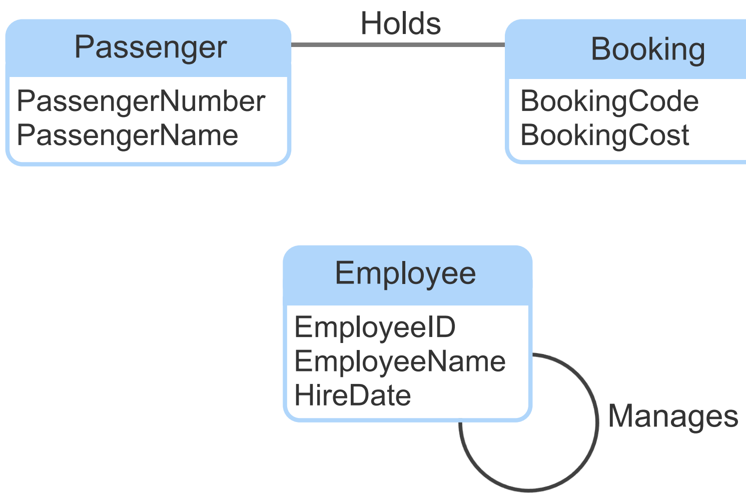
An ***entity-relationship model*** is a high-level representation of data requirements, ignoring implementation details. An entity-relationship model guides implementation in a particular database system, such as MySQL

An entity-relationship model includes three kinds of objects:

- An ***entity*** is a person, place, product, concept, or activity.
- A ***relationship*** is a statement about two entities.
- An ***attribute*** is a descriptive property of an entity.

A relationship is usually a statement about two different entities, but the two entities may be the same. A ***reflexive relationship*** relates an entity to itself.

PARTICIPATION ACTIVITY | 3.2.1: Entities, relationships, and attributes.



Animation content:

Step 1: In an airline reservation system, Passenger and Booking are entities. There are two entities named Passenger and Booking. An entity is represented by a colored box with the name inside.

Step 2: Holds is a relationship between Passenger and Booking. A line named Holds connects the two entities.

Step 3: PassengerNumber, PassengerName, BookingCode, BookingCost are attributes. Two rows are added to the Passenger entity as attributes with values Passenger Number and Passenger Name. Two rows are added to the Booking entity as attributes with values Booking Code and Booking Cost.

Step 4: In a human resources systems, Employee-Manages-Employee is a reflexive relationship. An entity Employee appears, with attributes EmployeeID, EmployeeName, and HireDate. A line named Manages connects Employee to itself.

Animation captions:

1. In an airline reservation system, Passenger and Booking are entities.
2. Holds is a relationship between Passenger and Booking.
3. PassengerNumber, PassengerName, BookingCode, BookingCost are attributes.
4. In a human resources systems, Employee-Manages-Employee is a reflexive relationship.

When the model is implemented in SQL, entities typically become tables. Relationships and attributes typically become foreign keys and columns, respectively. However, some relationships and attributes become tables. Since the conversion is indirect, requirements are documented as entities, relationships, and attributes rather than tables, keys, and columns.

Terminology

Attribute *is used in both entity-relationship and relational models. In the relational model, attribute is a formal term for column. Since entity-relationship attributes typically become relational columns, the meaning of attribute is similar in both models.*

PARTICIPATION ACTIVITY 3.2.2: Entities, relationships, and attributes.

How are the following requirements represented in an entity-relationship model?

The Widgets and Digits company has projects that are composed of multiple tasks. Projects have a limited duration, and tasks have a starting date.

1) A project is a(n) ________.

Check Show answer

2) The duration of a project is a(n) ________.

Check Show answer

3) "Task belongs to project" is a(n) ________.

Check Show answer

4) The starting date of a task is a(n) ________.

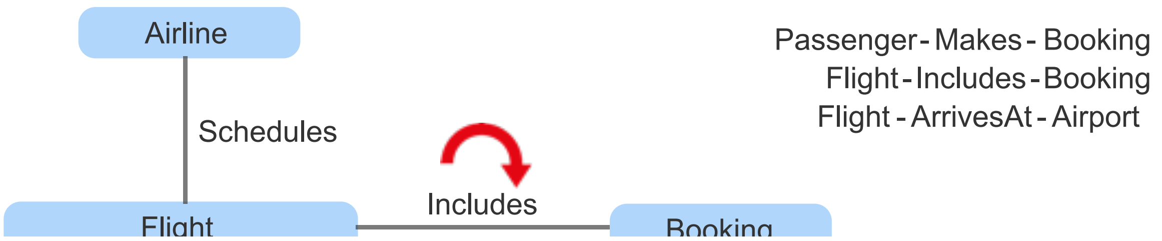
Check Show answer

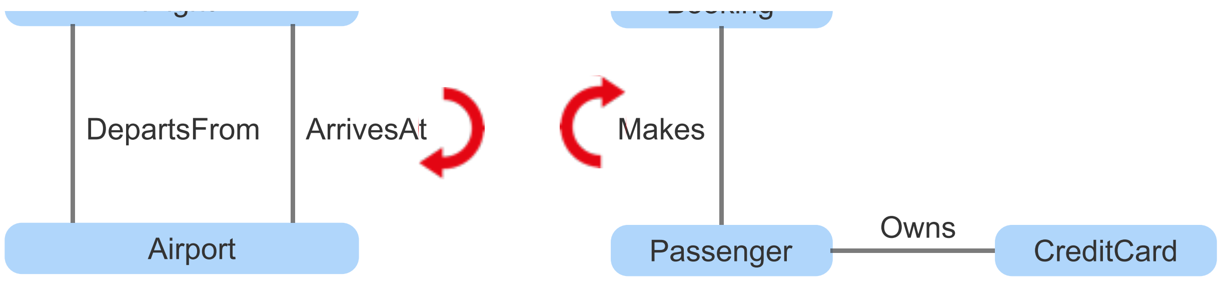
Entity-relationship diagram and glossary

Entities, relationships, and attributes are depicted in an ***entity-relationship diagram***, commonly called an ***ER diagram***. An ER diagram depicts entities as rectangles with rounded corners, relationships as lines connecting rectangles, and attributes within entity rectangles. ER diagrams always include entities and relationships. Attributes are optional and only appear when additional detail is needed.

The full name of a relationship includes the related entities and is written "Entity-Relationship-Entity". The full name is read clockwise around the center of the relationship, as illustrated in the animation below.

PARTICIPATION ACTIVITY | 3.2.3: The clockwise rule.


Airline
Schedules
Includes
Flight
Booking
Passenger - Makes - Booking
Flight - Includes - Booking
Flight - ArrivesAt - Airport



Animation content:

Step 1: This ER diagram of an airline reservation system has entities and relationships but no attributes. There are six entities named Airline Flight Airport Booking Passenger and Credit Card. A line named Schedules connects entities Airline and Flight. Two lines named Departs From and Arrives At connect entities Flight and Airport. A line named Includes connects entities Flight and Booking. A line Makes connects entities Booking and Passenger. A line named Owns connects entities Passenger and Credit Card.

Step 2: The full relationship name Passenger-Makes-Booking is read clockwise, around the center of the relationship line. A red arrow pointing up going from entity Passenger to entity Booking appears next to line Makes. Text duplicates and moves above from line Makes and entities Booking and Passenger to create the text Passenger hyphen Makes hyphen Booking.

Step 3: All relationships are read following the clockwise rule. A red arrow pointing right going from entity Flight to entity Booking appears next to line Includes. Text duplicates and moves above from line Includes and entities Flight and Booking to create the text Flight hyphen Includes hyphen Booking. A red arrow pointing down going from entity Flight to entity Airport appears next to line Arrives At. Text duplicates and moves above from line Arrives At and entities Flight and Airport to create the text Flight hyphen Arrives At hyphen Airport.

Animation captions:

1. This ER diagram of an airline reservation system has entities and relationships but no attributes.
2. The full relationship name Passenger-Makes-Booking is read clockwise, around the center of the relationship line.
3. All relationships are read following the clockwise rule.

A ***glossary***, also known as a ***data dictionary*** or ***repository***, documents additional detail in text format. A glossary includes names, synonyms, and descriptions of entities, relationships, and attributes. For simple databases with few users, a database designer may record the glossary with a text editor. For more complex databases, the designer may use a database or software tool specifically designed for glossaries.

The ER diagram and glossary are complementary and, together, completely describe an entity-relationship model.

PARTICIPATION ACTIVITY 3.2.4: ER diagram and glossary.

1) Referring to the animation above, how is the "schedules" relationship read?

- ○ Flight-Schedules-Airline
- ○ Airline-Schedules-Flight
- ○ Flight-IsScheduledBy-Airline

2) An entity-relationship model is completely described by:

- ○ An ER diagram
- ○ A glossary
- ○ Both an ER diagram and a glossary

3) What is included in a glossary?

- ○ Descriptions only
- ○ Names and synonyms only
- ○ Names, synonyms, and descriptions only
- ○ Names, synonyms, descriptions, and ER diagrams

Types and instances

In entity-relationship modeling, a type is a set:

- An ***entity type*** is a set of things. Ex: All employees in a company.
- A ***relationship type*** is a set of related things. Ex: Employee-Manages-Department is a set of (employee, department) pairs, where the employee manages the department.
- An ***attribute type*** is a set of values. Ex: All employee salaries.

Entity, relationship, and attribute types usually become tables, foreign keys, and columns, respectively.

An instance is an element of a set:

- An ***entity instance*** is an individual thing. Ex: The employee Sam Snead.
- A ***relationship instance*** is a statement about entity instances. Ex: "Maria Rodriguez manages Sales."
- An ***attribute instance*** is an individual value. Ex: The salary $35,000.

Entity, relationship, and attribute instances usually become rows, foreign key values, and column values, respectively

Table 3.2.1: Example types and instances.

	Type	Instance
Entity	Passenger	Muhammed Ali
Attribute	BookingCode	39240
Relationship	Passenger-Holds-Booking	Muhammed Ali holds 39240

PARTICIPATION ACTIVITY 3.2.5: Types and instances.

Match the term to the example.

If unable to drag and drop, refresh the page.

relationship instance **entity type** **relationship type** **entity instance**

attribute type **attribute instance**

	Students at San Antonio Community College
	Eleanor Rigby, a student at San Antonio community college

	Students take exams
	Eleanor Rigby takes the final exam in calculus
	Student record number
	324A21

Reset

Database design

Complex databases are developed in three phases:

1. ***Analysis*** develops an entity-relationship model, capturing data requirements while ignoring implementation details.
2. ***Logical design*** converts the entity-relationship model into tables, columns, and keys for a particular database system.
3. ***Physical design*** adds indexes and specifies how tables are organized on storage media.

Analysis is particularly important for complex databases with many users when documenting requirements is challenging. For small databases with just a few tables and users, analysis is less important and often omitted.

Analysis and logical design steps are summarized in the table below. Although these steps are presented in sequence, in practice execution is not always sequential. Often an early step is revisited after a later step is completed.

Physical design is dependent on specific index and table structures, which vary greatly across relational databases. Physical design is discussed elsewhere in this material.

Table 3.2.2: Analysis steps.

Step	Name
1	Discover entities, relationships, and attributes
2	Determine cardinality
3	Distinguish strong and weak entities

4	Create supertype and subtype entities

Table 3.2.3: Logical design steps.

Step	Name
5	Implement entities
6	Implement relationships
7	Implement attributes
8	Apply normal form

PARTICIPATION ACTIVITY 3.2.6: Analysis and logical design.

1) Analysis considers implementation issues related to a specific database system.

- ○ True
- ○ False

2) An entity-relationship model is developed for all database design projects.

- ○ True
- ○ False

3) Entities, relationships, and attributes always map directly to tables, foreign keys, and columns, respectively.

- ○ True
- ○ False

3.3 Cardinality

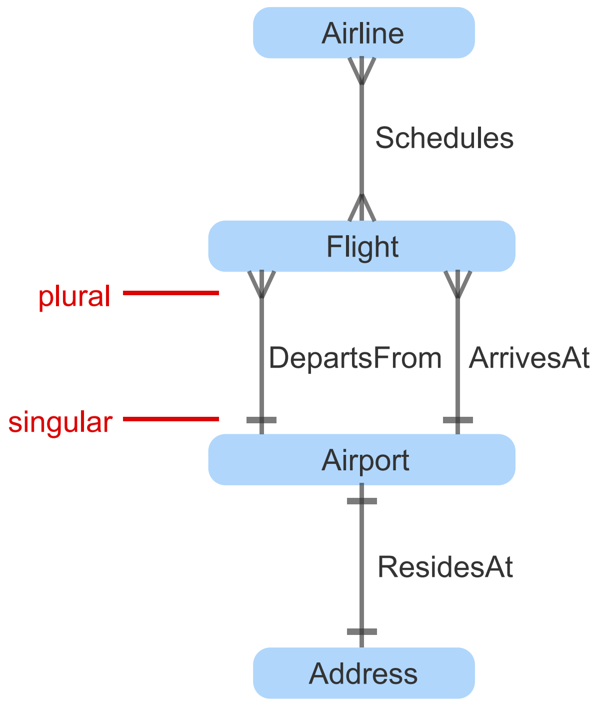
Relationship maximum

In entity-relationship modeling, ***cardinality*** refers to maxima and minima of relationships and attributes.

Relationship maximum is the greatest number of instances of one entity that can relate to a single instance of another entity. A relationship has two maxima, one for each of the related entities. Maxima are usually specified as one or many. A related entity is ***singular*** when the maximum is one and ***plural*** when the maximum is many.

On ER diagrams, maximum of one is shown as a short bar across the relationship line. Maximum of many is shown as three short lines that converge at a point. The three lines look like a bird's foot, so this convention is called ***crow's foot*** notation.

PARTICIPATION ACTIVITY 3.3.1: Relationship maximum.



Animation content:

Step 1: Each flight departs from at most one airport. Airport is singular in Flight-DepartsFrom-

Airport. There are four entities named Airline, Flight, Airport, and Address. A line Schedules connects Airline and Flight. Lines ArrivesAt and DepartsFrom connect Flight and Airport. A line ResidesAt connects Airport and Address. A short line across line DepartsFrom appears above Airport. The caption singular appears next to the short line.

Step 2: Each airport has many departing flights. Flight is plural in Flight-DepartsFrom-Airport. Along line DepartsFrom, the crow's foot symbol appears below entity Flight. The caption plural appears next to the crow's foot symbol.

Step 3: ArrivesAt and DepartsFrom have the same maxima. Along the ArrivesAt line, a crow's foot symbol appears below Flight and a short line appears above Airport.

Step 4: Each airline schedules many flights. When multiple airlines share the same flights, each flight can be scheduled by many airlines. Along the Schedules line, the crow's foot symbol appears below Airline and above Flight.

Step 5: Each airport resides at most one official address. Each address has at most one airport. Along the ResidesAt line, a short line appears below Airport and above Address.

Animation captions:

1. Each flight departs from at most one airport. Airport is singular in Flight-DepartsFrom-Airport.
2. Each airport has many departing flights. Flight is plural in Flight-DepartsFrom-Airport.
3. ArrivesAt and DepartsFrom have the same maxima.
4. Each airline schedules many flights. When multiple airlines share the same flights, each flight can be scheduled by many airlines.
5. Each airport resides at most one official address. Each address has at most one airport.

Occasionally, a plural entity has a fixed numeric maximum. Ex: In Employee-Has-Telephone, if each employee has at most three telephone numbers, the maximum of Telephone is three. Fixed numeric maxima are documented in the glossary.

PARTICIPATION ACTIVITY 3.3.2: Relationship maximum.

Determine the maxima for each relationship. The correct answer may depend on business rules.

1) Student-Takes-Course

○ one-one

- ○ one-many
- ○ many-one
- ○ many-many

2) City-IsCapitalOf-State

- ○ one-one
- ○ one-many
- ○ many-one
- ○ many-many

3) Airport-Has-Runway

- ○ one-one
- ○ one-many
- ○ many-one
- ○ many-many

4) Person-Owns-Vehicle

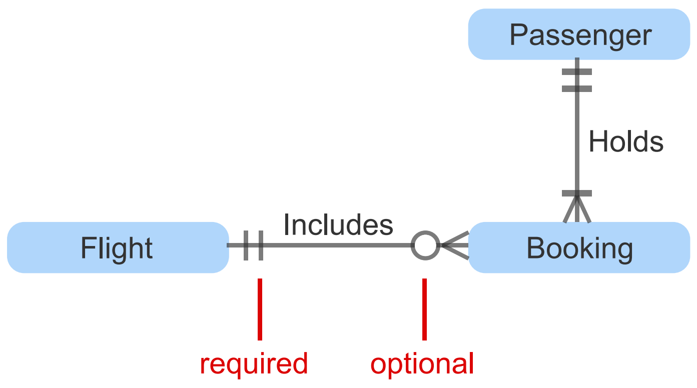
- ○ one-one
- ○ one-many
- ○ many-one
- ○ many-many

Relationship minimum

Relationship minimum is the least number of instances of one entity that can relate to a single instance of another entity. A relationship has two minima, one for each of the related entities. Minima are usually specified as zero or one. A related entity is ***optional*** when the minimum is zero and ***required*** when the minimum is one.

On ER diagrams, minimum of one is shown as a short bar across the relationship line. Minimum of zero is shown as a circle. Maxima symbols always appear next to the entity and minima symbols appear further from the entity.

PARTICIPATION ACTIVITY | 3.3.3: Relationship minimum.



Animation content:

Step 1: The maximum of Flight-Includes-Booking is one-many. There are three entities named Passenger, Booking, and Flight. A line named Holds connects Passenger and Booking. A line named Includes connects Flight and Booking. Along the Includes line, a short line appears next to Flight and a crow's foot symbol appears next to Booking.

Step 2: Each booking is included on at least one flight. Flight is required in Flight-Includes-Booking. Across the Includes line, a second short line, with caption required, appears next to Flight.

Step 3: A new flight may have no bookings. Booking is optional in Flight-Includes-Booking. Across the Includes line, a circle with caption optional appears next to Booking.

Step 4: The maximum of Passenger-Holds-Booking is one-many. Along the Holds line, a short line appears next to Passenger and a crow's foot symbol appears next to Booking.

Step 5: Passenger and Booking are both required in Passenger-Holds-Booking. Across the Holds line, short lines appear next to Passenger and Booking.

Animation captions:

1. The maximum of Flight-Includes-Booking is one-many.
2. Each booking is included on at least one flight. Flight is required in Flight-Includes-Booking.
3. A new flight may have no bookings. Booking is optional in Flight-Includes-Booking.
4. The maximum of Passenger-Holds-Booking is one-many.
5. Passenger and Booking are both required in Passenger-Holds-Booking.

Occasionally, a required entity has a minimum greater than one. Ex: In Customer-Has-Identification, if two forms of identification are required for every customer, the minimum of Identification is two

Minima greater than one are documented in the glossary.

PARTICIPATION ACTIVITY 3.3.4: Relationship minimum.

Determine the minima for each relationship. The correct answer may depend on business rules.

1) Person-Marries-Person

- zero-zero
- zero-one
- one-zero
- one-one

2) Person-Has-Passport

- zero-zero
- zero-one
- one-zero
- one-one

3) Flight-ArrivesAt-Airport

- zero-zero
- zero-one
- one-zero
- one-one

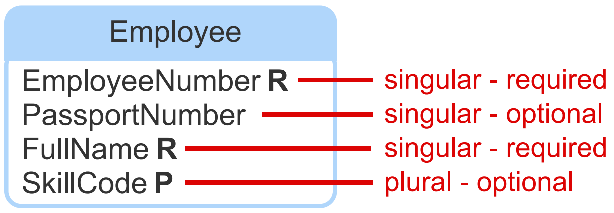
Attribute maximum and minimum

Attribute maximum is the greatest number of attribute values that can describe each entity instance. Attribute maximum is specified as one (singular) or many (plural).

Attribute minimum is the least number of attribute values that can describe each entity instance. Attribute minimum is specified as zero (optional) or one (required).

In ER diagrams, attributes are presumed singular and optional. A "P" following the attribute indicates the attribute is plural. An "R" indicates the attribute is required.

PARTICIPATION ACTIVITY 3.3.5: Attribute maximum and minimum.



Animation content:

Step 1: Each employee has exactly one employee number. There is an entity named Employee with attributes EmployeeNumber, PassportNumber, FullName, and SkillCode. R appears after EmployeeNumber and is labeled singular-required.

Step 2: Each employee has at most one United States passport number. Some employees do not have a passport. PassportNumber is labeled singular-optional.

Step 3: Each employee has exactly one name. R appears after FullName and is labeled singular-required.

Step 4: An employee may have many skills. Some employees record no skills in the database. P appears after SkillCode and is labeled plural-optional.

Animation captions:

1. Each employee has exactly one employee number.
2. Each employee has at most one United States passport number. Some employees do not have a passport.
3. Each employee has exactly one name.
4. An employee may have many skills. Some employees record no skills in the database.

Occasionally, attribute maximum or minimum is a fixed number other than zero, one, or many. Ex: If each employee must speak at least two languages, the Language attribute of Employee has a minimum of two. Numeric minimum and maximum are documented in the glossary.

PARTICIPATION ACTIVITY 3.3.6: Attribute maximum and minimum.

Determine the maximum and minimum of each attribute. The correct answer may depend on business rules.

1) Entity Airport, attribute AirportCode

- ○ singular-required
- ○ singular-optional
- ○ plural-required

2) Entity Person, attribute SpouseName

- ○ singular-required
- ○ plural-required
- ○ singular-optional

3) Entity Employee, attribute LanguageCode

- ○ singular-required
- ○ plural-optional
- ○ plural-required

Unique attributes

Each value of a ***unique attribute*** describes at most one entity instance. Ex: Vehicle identification number (VIN) is a unique attribute of Vehicle.

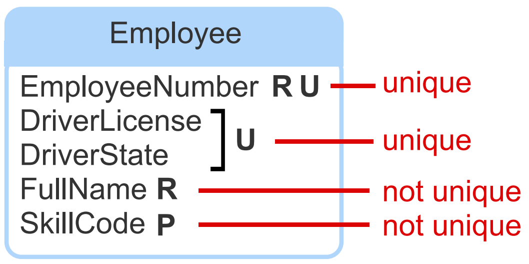
Occasionally, a composite of several attributes is unique, although the individual attributes are not. Ex: (AirlineCode, FlightNumber) is a unique composite attribute of Flight. However, AirlineCode and FlightNumber are not individually unique, since flights on different airlines may have the same flight number.

Unique is not the same as a singular:

- A unique attribute has at most one entity instance for each attribute value.
- A singular attribute has at most one attribute value for each entity instance.

In ER diagrams, attributes are presumed not unique. A "U" following the attribute indicates the attribute is unique. Unique composite attributes are grouped with a brace before the "U" symbol.

PARTICIPATION ACTIVITY 3.3.7: Unique attributes.



Animation content:

Step 1: Each employee number describes at most one employee. An entity named Employee has attributes EmployeeNumber R, DriverLicense, DriverState, FullName R, and SkillCode P. U, with caption unique, appears next to EmployeeNumber R.

Step 2: License number and state are not separately unique, but the composite is unique. One bracket appears next to attributes DriverLicense and DriverState. U, with caption unique, appears next to the bracket.

Step 3: Each name and each skill can describe many employees. Caption not unique appears next to FullName and SkillCode.

Step 4: Unique is not the same as singular. Ex: FullName is singular but not unique. FullName is highlighted.

Animation captions:

1. Each employee number describes at most one employee.
2. License number and state are not separately unique, but the composite is unique.
3. Each name and each skill can describe many employees.
4. Unique is not the same as singular. Ex: FullName is singular but not unique.

Entity-Has-Attribute relationship

Entities have an implicit relationship with their attributes, called Entity-Has-Attribute. Attribute maximum and minimum are the cardinality of Attribute in this relationship. Ex: VIN is a required attribute of Vehicle. In the relationship Vehicle-Has-VIN, every vehicle must have a VIN..

An attribute is unique when Entity is singular in this relationship. Ex: Each VIN has at most one vehicle. So VIN is a unique attribute of Vehicle.

PARTICIPATION ACTIVITY | 3.3.8: Unique attributes.

Match the term to the database requirement.

If unable to drag and drop, refresh the page.

unique plural required optional singular

	We track at most one contact telephone number for each student. TelephoneNumber is a(n) ______ attribute of Student.
	Students can major in several subjects. MajorSubjectName is a(n) ______ attribute of Student.
	Sometimes students do not provide a telephone number on registration forms, so this information is left blank in the database. TelephoneNumber is a(n) ______ attribute of Student.
	All students must have an official email address. EmailAddress is a(n) ______ attribute of Student.
	Students are assigned an eight-digit number, used to identify students on forms and records. StudentNumber is a(n) ______ attribute of Student.

Reset

Database design

Relationship and attribute cardinality depends on business rules. Ex: Usually Employee-WorksIn-Department maxima are many-one. If a company assigns employees to multiple departments, however, the maxima are many-many.

During the analysis phase, the designer looks for cardinality business rules in interviews and document review. The designer then converts business rules into zero, one, and many specifications, and documents specifications in the ER diagram and glossary.

Depending on the desired level of detail, cardinality does not always appear on ER diagrams. Usually, ER diagrams are drawn with software tools that can automatically show or hide cardinality.

Table 3.3.1: Determine cardinality.

Step	Activity
2A	Determine relationship maxima and minima.
2B	Determine attribute maxima and minima.
2C	Identify unique attributes.
2D	Document cardinality in glossary and, optionally, on ER diagram.

PARTICIPATION ACTIVITY 3.3.9: Database design.

1) 'Cardinality' refers to relationships only.

- True
- False

2) ER diagrams indicate maxima and minima for relationships only.

- True
- False

3) Maxima and minima usually depend

on business rules.

- ○ True
- ○ False

4) In the analysis phase, cardinality is always determined after discovery is complete.

- ○ True
- ○ False

CHALLENGE ACTIVITY 3.3.1: Cardinality.

544874.3500394.qx3zqy7

Start

Hundreds of students are in one course.
Each student takes multiple courses.

Determine the relationship maxima.

Student-Takes-Course

Select

1	2	3	4	5

Check Next

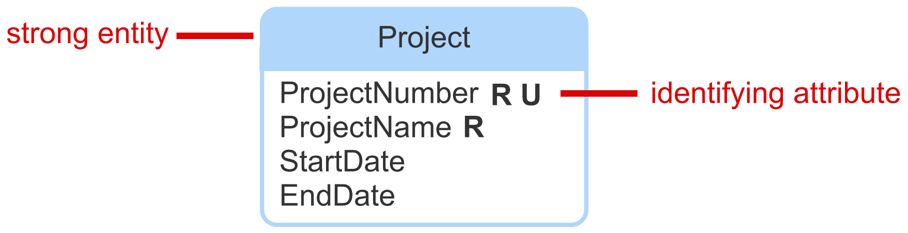
3.4 Strong and weak entities

Strong entities

An ***identifying attribute*** is unique, singular, and required. Identifying attribute values correspond one-to-one to, or ***identify***, entity instances.

A ***strong entity*** has one or more identifying attributes. When a strong entity is implemented as a table, one of the identifying attributes may become the primary key.

PARTICIPATION ACTIVITY | 3.4.1: Strong entities.



Animation content:

Static figure:
The Project entity has attributes ProjectNumber R U, ProjectName R, StartDate, and EndDate. The Project entity has caption strong entity. The ProjectNumber attribute has caption identifying attribute.

Step 1: Each number describes at most one project, so ProjectNumber is unique. The U following ProjectNumber is highlighted.

Step 2: Each project has exactly one number, so ProjectNumber is singular and required. The R following ProjectNumber is highlighted.

Step 3: ProjectNumber is an identifying attribute. Project numbers correspond one-to-one to projects. The caption identifying attribute appears next to ProjectNumber.

Step 4: An entity with an identifying attribute is strong. The caption strong entity appears next to Project.

Animation captions:

1. Each number describes at most one project, so ProjectNumber is unique.

1. Each number describes at most one project, so ProjectNumber is unique.
2. Each project has exactly one number, so ProjectNumber is singular and required.
3. ProjectNumber is an identifying attribute. Project numbers correspond one-to-one to projects.
4. An entity with an identifying attribute is strong.

PARTICIPATION ACTIVITY | 3.4.2: Identifying attributes.

The Project entity has a ProjectCode attribute.

1) Each project has at most one code. The ProjectCode attribute is _____.

- unique
- singular
- plural
- required
- optional

2) Each project code describes at most one project. The ProjectCode attribute is _____.

- unique
- singular
- plural
- required
- optional

3) A project may have no code. The ProjectCode attribute is _____.

- unique
- singular
- plural
- required
- optional

4) ProjectCode is an identifying attribute

of the Project entity.

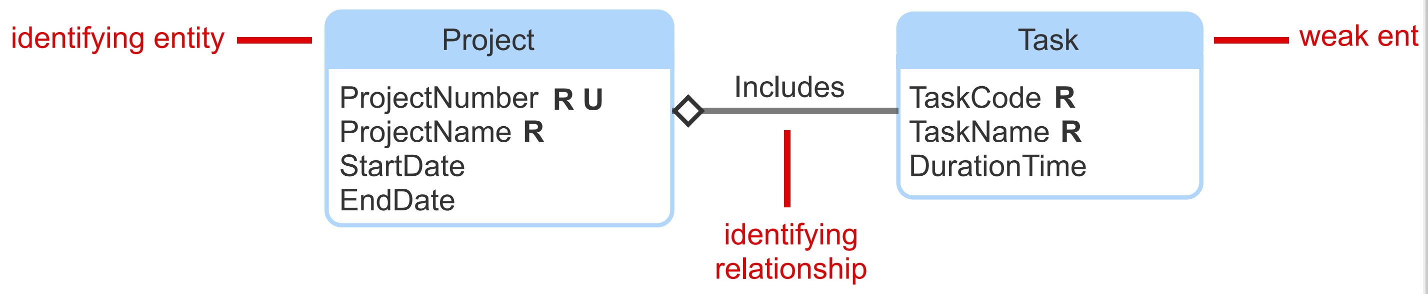
- ○ True
- ○ False

Weak entities

A ***weak entity*** does not have an identifying attribute. Instead, a weak entity usually has a relationship, called an ***identifying relationship***, to another entity, called an ***identifying entity***. The identifying entity must be singular and required in an identifying relationship.

In an ER diagram, an identifying relationship has a diamond next to the identifying entity. Since an identifying entity is always singular and required, the diamond replaces the entity's cardinality symbols.

PARTICIPATION ACTIVITY | 3.4.3: Weak entities.



Animation content:

Static figure:
The Project entity has attributes ProjectNumber R U, ProjectName R, StartDate, and EndDate. The Project entity has caption identifying entity.

The Task entity has attributes TaskCode R, TaskName R, DurationTime. The Task entity has caption weak entity.

The relationship Project-Includes-Task has a diamond next to Project. The relationship has caption identifying relationship.

Step 1: TaskCode, TaskName, and DurationTime are not unique. Since Task does not have an identifying attribute, Task is a weak entity. Task appears with caption weak entity.

Step 2: Task is weak and Project is singular and required, so Project-Includes-Task is an identifying relationship. Project appears. The Includes relationship appears with caption identifying relationship. Along the Includes line, two short lines appear next to Project.

Step 3: A diamond indicates an identifying relationship. The diamond is next to the identifying entity and replaces the cardinality symbols. The two short lines are replaced with a diamond. The caption identifying entity appears next to Project.

Animation captions:

1. TaskCode, TaskName, and DurationTime are not unique. Since Task does not have an identifying attribute, Task is a weak entity.
2. Task is weak and Project is singular and required, so Project-Includes-Task is an identifying relationship.
3. A diamond indicates an identifying relationship. The diamond is next to the identifying entity and replaces the cardinality symbols.

For weak entities, identifying relationships replace identifying attributes. Ex: In the animation above, If each project has at most one task, ProjectNumber identifies Task. If each project has many tasks, (ProjectNumber, TaskName) identifies Task. The second attribute, TaskName, must be singular, required, and unique within each project.

PARTICIPATION ACTIVITY 3.4.4: Strong and weak entities.

Refer to the following database requirements:

A model has entities Department, Course, and Exam. Each academic department has a unique name.

Courses are offered by academic departments. Courses are identified by a course number and the name of the department that offers the course. Different departments use the same course numbers. Ex: Math 101, Philosophy 101, Economics 200A.

Each course has several exams. Each exam is labeled with a course name and exam sequence number. The sequence number begins at 1 for all courses. Ex: Math 101-1, Math 101-2, Math 101-3.

If unable to drag and drop, refresh the page.

Identifying entity | Identifying attribute | Identifying relationship | Strong entity | Weak entity

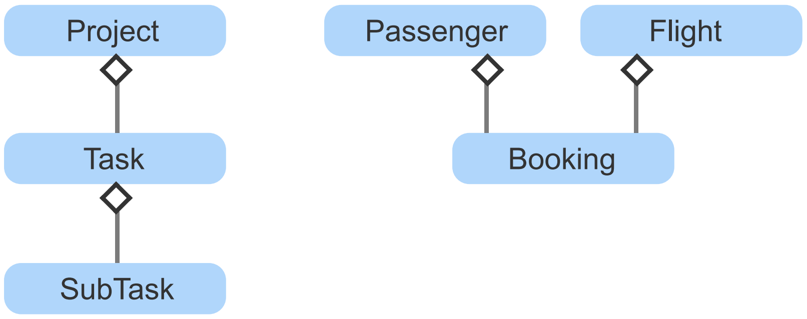
	DepartmentName
	Course, Exam
	Department
	Department, Course
	Department-Offers-Course, Course-Gives-Exam

Reset

Identifying entities

A weak entity is usually identified by a strong entity. However, a weak entity can be identified by another weak entity or by several entities.

PARTICIPATION ACTIVITY | 3.4.5: Identifying entities.



Animation content:

Static figure:
Entities Project, Task, and SubTask appear. Project and Task are connected by an unnamed relationship, with a diamond next to Project. Task and SubTask are connected by an unnamed relationship, with a diamond next to Task.

Entities Passenger, Flight, and Booking appear. Passenger and Booking are connected by an unnamed relationship, with a diamond next to Passenger. Flight and Booking are connected by an unnamed relationship, with a diamond next to Flight.

Step 1: Project has an identifying attribute, such as ProjectNumber, and is a strong entity. The Project entity appears.

Step 2: Task is a weak entity, identified by the Project entity. The Task entity appears. The unnamed relationship between Project and Task appears with a diamond next to Project.

Step 3: The weak entity Subtask is identified by the weak entity Task. The SubTask entity appears. The unnamed relationship between Task and SubTask appears with a diamond next to Task.

Step 4: The weak entity Booking is identified by Passenger and Flight. Passenger, Flight, and Booking appear, along with the associated relationships and diamonds.

Animation captions:

1. Project has an identifying attribute, such as ProjectNumber, and is a strong entity.
2. Task is a weak entity, identified by the Project entity.
3. The weak entity Subtask is identified by the weak entity Task.
4. The weak entity Booking is identified by Passenger and Flight.

When a weak entity is identified by a weak entity or multiple entities, the identifying attribute may be complex. Ex: In the animation above:

- If each task has many subtasks, a composite attribute such as (ProjectNumber, TaskName, SubtaskCode) identifies Subtask.
- If each person and each flight has many bookings, a composite attribute such as (FlightNumber, PassengerID, BookingDate) identifies Booking.

In these cases, the identifying attribute depends on business rules and may not be apparent in the ER diagram.

PARTICIPATION ACTIVITY 3.4.6: Identifying entities.

1) How often is an identifying entity also a weak entity?

- ○ Never
- ○ Sometimes
- ○ Always

2) How many relationships can identify one weak entity?

- ○ Zero or one
- ○ Exactly one
- ○ One or many

3) How many weak entities can one entity identify?

- ○ Zero or one
- ○ One or many
- ○ Zero, one, or many

Database design

After entities, relationships, attributes, and cardinality are determined, the database designer distinguishes strong and weak entities. For each weak entity, the identifying relationship is noted. Weak entities and identifying relationships are documented in the glossary and ER diagram.

Most database designers use software tools to manage ER diagrams. Software tools usually allow users to choose from alternative conventions and automatically switch between conventions.

Table 3.4.1: Distinguish strong and weak entities.

Step	Activity
3A	Identify strong and weak entities.
3B	Determine the identifying relationship(s) for each weak entity.
3C	Document weak entities and identifying relationships in glossary and ER diagram.

PARTICIPATION ACTIVITY 3.4.7: Distinguish strong and weak entities.

1) Distinguishing strong and weak entities is a logical design activity.

- ○ True
- ○ False

2) Determining cardinality always precedes distinguishing strong and weak entities.

- ○ True
- ○ False

3) Database designers may look for identifying relationships first, before noting weak entities.

- ○ True
- ○ False

CHALLENGE ACTIVITY 3.4.1: Strong and weak entities.

544874.3500394.qx3zqy7

Start

A database tracks municipal construction projects. Each project has a different code and ı orders for materials. Each order is named with the project code and an order number, begi 1000 for all projects. Orders have one or more line items, numbered sequentially, beginnin orders.

Select the correct identifying entity.

Project	Select
Order	Select
LineItem	Select

1 2

Check Next

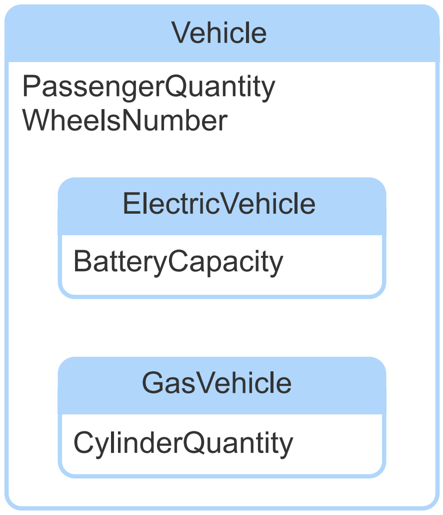
3.5 Supertype and subtype entities

Supertype and subtype entities

An entity type is a set of entity instances. A ***subtype entity*** is a subset of another entity type, called the ***supertype entity***. Ex: Managers are a subset of employees, so Manager is a subtype entity of the Employee supertype entity. On ER diagrams, subtype entities are drawn within the supertype.

A supertype entity usually has several subtypes. Attributes of the supertype apply to all subtypes. Attributes of a subtype do not apply to other subtypes or the supertype.

PARTICIPATION ACTIVITY 3.5.1: Supertype and subtype entities.



Animation content:

Step 1: Vehicle supertype has ElectricVehicle and GasVehicle subtypes. Entity Vehicle appears. Entities ElectricVehicle and GasVehicle appear inside entity Vehicle.

Step 2: Number of passengers and number of wheels apply to both electric and gas vehicles.

Attributes PassengerQuantity and WheelsNumber appear in Vehicle.

Step 3: Battery capacity applies to electric vehicles only. Attribute BatteryCapacity appears in ElectricVehicle.

Step 4: Number of cylinders applies to gas vehicles only. Attribute CylinderQuantity appears in GasVehicle.

Animation captions:

1. Vehicle supertype has ElectricVehicle and GasVehicle subtypes.
2. Number of passengers and number of wheels apply to both electric and gas vehicles.
3. Battery capacity applies to electric vehicles only.
4. Number of cylinders applies to gas vehicles only.

Supertype and subtype entities have several benefits:

- Subtypes highlight subsets of data with different attributes and relationships than the supertype. This clarifies database semantics and behavior.
- Optional supertype attributes become required subtype attributes. This results in compact tables with fewer NULL values.
- An attribute that is repeated in several subtypes becomes a single supertype attribute. This ensures a single name, description, cardinality, and type for the attribute, rather than one for each subtype.

Examples of optional and repeated attributes appear in the subsection *Similar entities and optional attributes*, below.

PARTICIPATION ACTIVITY | 3.5.2: Supertype and subtype entities.

Refer to the following database requirements:

Some passengers enroll in mileage plans and get a mileage plan number. Travelers who fly 100,000 miles or more in a calendar year are gold mileage plan members. Between 25,000 miles and 100,000 miles is silver status, and below 25,000 miles flown is bronze status. Gold members designate the name of a family member who can fly for free.

1) What is passenger?

- ○ Supertype entity
- ○ Subtype entity

- ○ Weak entity

2) What is gold plan member?
   - ○ Supertype entity
   - ○ Subtype entity
   - ○ Attribute

3) What is mileage plan number?
   - ○ Subtype entity
   - ○ Attribute of Passenger
   - ○ Attribute of GoldPlanMember

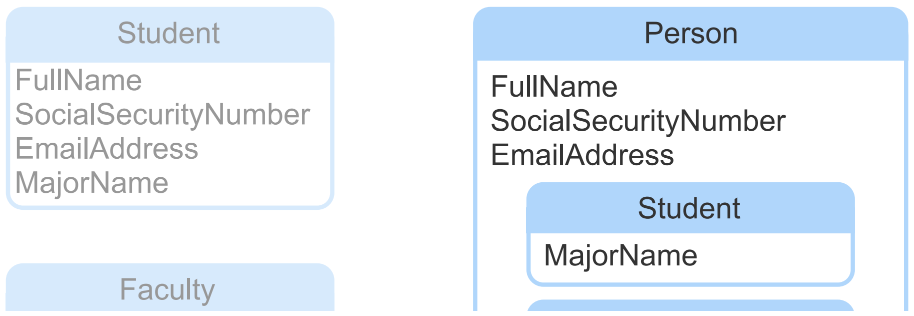
4) What is the name of the family member designated by gold status passengers?
   - ○ Subtype entity
   - ○ Attribute of Passenger
   - ○ Attribute of GoldPlanMember

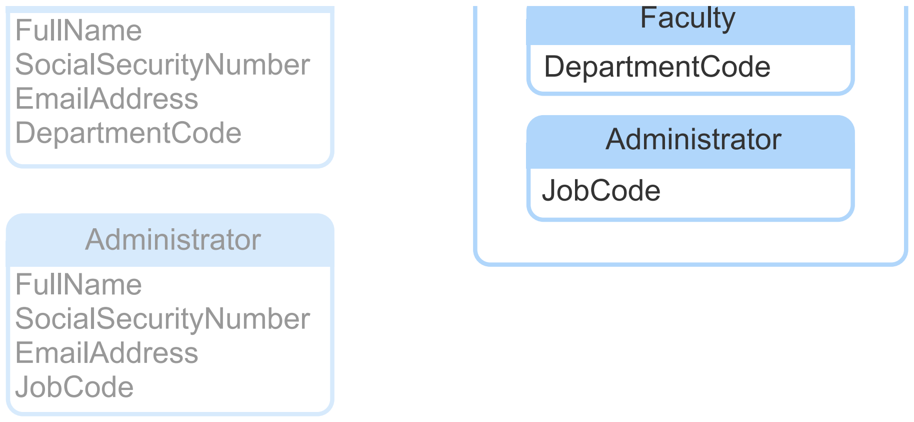
Similar entities and optional attributes

Supertype and subtype entities are often created from similar entities and optional attributes.

Similar entities are entities that have many common attributes and relationships. Similar entities become subtypes of a new supertype entity, as in the animation below. Common attributes and relationships move to the new supertype entity. Attributes and relationships that are not shared remain with the subtype entities.

PARTICIPATION ACTIVITY | 3.5.3: Similar entities become supertype entity.





Animation content:

Static figure:
Two entity-relationship diagrams appear.

In the first diagram, entity Student has attributes FullName, SocialSecurityNumber, EmailAddress, and MajorName. Entity Faculty has attributes FullName, SocialSecurityNumber, EmailAddress, and DepartmentCode. Entity Administrator has attributes FullName, SocialSecurityNumber, EmailAddress, and JobCode The first diagram is faded out.

In the second diagram, entity Person has attributes FullName, SocialSecurityNumber, and EmailAddress. Entities Student, Faculty, and Administrator are inside Person. Student has attribute MajorName. Faculty has attribute DepartmentCode. Administrator has attribute JobCode.

Step 1: Student, Faculty, and Administrator are similar entities with common attributes FullName, SocialSecurityNumber, and EmailAddress. The first diagram appears.

Step 2: Common attributes move to a new supertype entity called Person. In the first diagram, attributes FullName, SocialSecurityNumber, and EmailAddress are highlighted in all three entities. The Person entity appears with attributes FullName, SocialSecurityNumber, and EmailAddress.

Step 3: Subtype-specific attributes remain attributes of each subtype. In the second diagram, entities Student, Faculty, and Administrator appear inside Person. Student has attribute MajorName. Faculty has attribute DepartmentCode. Administrator has attribute JobCode. The first diagram fades out.

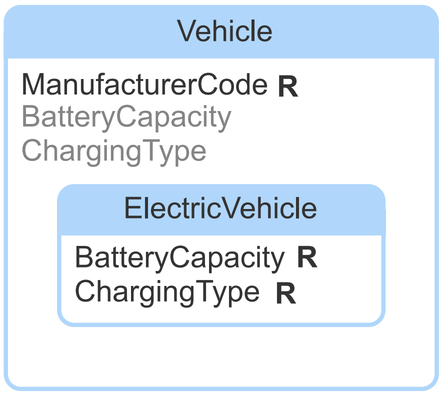
Animation captions:

1. Student, Faculty, and Administrator are similar entities with common attributes FullName, SocialSecurityNumber, and EmailAddress.

2. Common attributes move to a new supertype entity called Person.
3. Subtype-specific attributes remain attributes of each subtype.

An entity with many optional attributes also suggests new supertype and subtype entities. The entity becomes a supertype entity and retains all required attributes. Optional attributes become required attributes of new subtype entities.

PARTICIPATION ACTIVITY | 3.5.4: Optional attributes become subtype entities.



Animation content:

Step 1: Vehicle has required attribute ManufacturerCode and optional attributes BatteryCapacity and ChargingType. Entity Vehicle appears with attributes ManufacturerCode R, BatteryCapacity, and ChargingType.

Step 2: Optional attributes become required attributes of the new subtype entity ElectricVehicle. Entity ElectricVehicle appears inside Vehicle. Attributes BatteryCapacity and ChargingType move from Vehicle to ElectricVehicle. An R appears after both attributes.

Animation captions:

1. Vehicle has required attribute ManufacturerCode and optional attributes BatteryCapacity and ChargingType.
2. Optional attributes become required attributes of the new subtype entity ElectricVehicle.

Creating a new supertype for similar entities, or a new subtype for optional attributes, is neither an automatic nor objective decision. Similar entities with many common attributes are good candidates for a new supertype. Entities with many optional attributes are good candidates for a new subtype.

PARTICIPATION ACTIVITY | 3.5.5: Similar entities and optional attributes.

Refer to the following database requirements:

The university tracks the grade point average and telephone number for all students. The university also tracks the phone number for faculty and administrative staff. Faculty have an academic ranking, like "Assistant Professor" or "Adjunct Professor". Administrative staff are either hourly or salaried, which the university calls employment status.

The requirements are modeled as a supertype entity Person with subtypes Student, Faculty, and Administrator.

1) Which entity does GradePointAverage belong to?

- ○ Person
- ○ Student
- ○ Faculty
- ○ Administrator

2) Which entity does TelephoneNumber belong to?

- ○ Person
- ○ Student
- ○ Faculty
- ○ Administrator

3) Which entity does EmploymentStatus belong to?

- ○ Person
- ○ Student
- ○ Faculty
- ○ Administrator

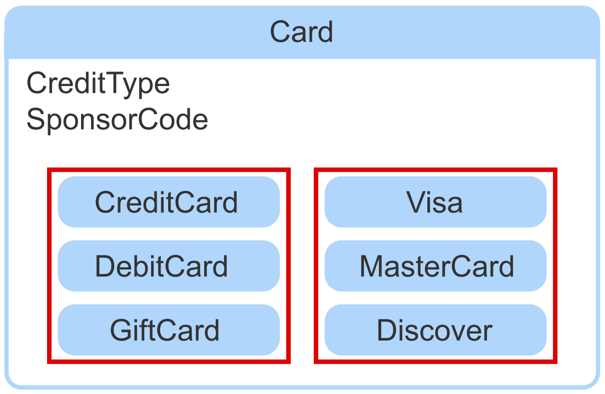
Partitions

A ***partition*** of a supertype entity is a group of mutually exclusive subtype entities. A supertype entity can have several partitions. Subtype entities within each partition are disjoint and do not share instances. Subtype entities in different partitions overlap and do share instances.

In diagrams, subtype entities within each partition are vertically aligned. Subtype entities in different partitions are horizontally aligned.

Each partition corresponds to an optional ***partition attribute*** of the supertype entity. The partition attribute indicates which subtype entity is associated with each supertype instance.

PARTICIPATION ACTIVITY | 3.5.6: Partitions.



Animation content:

Step 1: Subtypes CreditCard, DebitCard, and GiftCard are mutually exclusive. These subtypes partition the supertype Card. Entity Card appears. Entities CreditCard, DebitCard, and GiftCard appear inside Card, stacked vertically.

Step 2: CreditType is a partition attribute. Values are "Credit", "Debit", "Gift". Attribute CreditType appears in Card.

Step 3: Visa, MasterCard, and Discover are another partition of Card. The partition attribute is SponsorCode. Entities Visa, MasterCard, and Discover appear inside Card, stacked vertically. Attribute SponsorCode appears in Card.

Step 4: Card has two partitions. Subtypes within each partition are disjoint. Subtypes in different partitions overlap. A box surrounds CreditCard, DebitCard, and GiftCard. Another box surrounds entities Visa, MasterCard, and Discover.

entities Visa, MasterCard, and Discover.

Animation captions:

1. Subtypes CreditCard, DebitCard, and GiftCard are mutually exclusive. These subtypes partition the supertype Card.
2. CreditType is a partition attribute. Values are "Credit", "Debit", "Gift".
3. Visa, MasterCard, and Discover are another partition of Card. The partition attribute is SponsorCode.
4. Card has two partitions. Subtypes within each partition are disjoint. Subtypes in different partitions overlap.

PARTICIPATION ACTIVITY | 3.5.7: Partitions and partition attributes.

1) An entity instance can be in two subtypes of the same partition.
- ○ True
- ○ False

2) An entity instance can be in two subtypes of different partitions.
- ○ True
- ○ False

3) Each partition attribute value corresponds to one subtype.
- ○ True
- ○ False

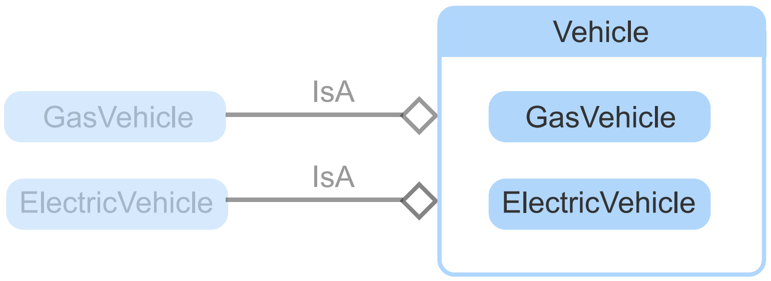
4) One partition attribute can correspond to several partitions.
- ○ True
- ○ False

IsA relationship

A supertype entity identifies its subtype entities. The identifying relationship is called an ***IsA relationship***. Ex: Manager-IsAn-Employee relates each manager instance to the corresponding employee instance. Employee numbers, which identify all employees, also identify managers.

Since every subtype has an IsA relationship to the supertype, the relationship is assumed and may be omitted from the ER diagram.

PARTICIPATION ACTIVITY 3.5.8: IsA relationship.



Animation content:

Step 1: Subtype entities have an identifying relationship, called IsA, to the supertype. Entity Vehicle appears. Entities GasVehicle and ElectricVehicle appear outside of Vehicle. Relationships GasVehicle-IsA-Vehicle and ElectricVehicle-IsA-Vehicle appear. Along both relationships, a diamond appears next to Vehicle.

Step 2: IsA relationships are not shown in an ER diagram. GasVehicle and ElectricVehicle move inside Vehicle. The IsA relationships disappear.

Animation captions:

1. Subtype entities have an identifying relationship, called IsA, to the supertype.
2. IsA relationships are not shown in an ER diagram.

PARTICIPATION ACTIVITY 3.5.9: IsA relationship.

1) Subtype entities must have an identifying attribute

True

False

2) The identifying attribute of a supertype entity also identifies the subtype entities.

True

False

3) All subtype entities are weak entities.

True

False

4) All weak entities are subtype entities.

True

False

Database design

After entities, relationships, attributes, cardinality, and strong and weak entities are determined, the database designer looks for supertype and subtype entities. Similar entities and optional attributes suggest new supertype and subtype entities and warrant special attention. Mutually exclusive subtype entities are grouped into partitions. For each partition, a partition attribute is added to the supertype entity.

Table 3.5.1: Create supertype and subtype entities.

Step	Activity
4A	Identify supertype and subtype entities.
4B	Replace similar entities and optional attributes with supertype and subtype entities.
4C	Identify partitions and partition attributes.
4D	Document supertypes, subtypes, and partitions in glossary and ER diagram.

Creating supertype and subtype entities is the last of four analysis steps:

1. Discover entities, relationships, and attributes
2. Determine cardinality
3. Distinguish strong and weak entities
4. Create supertype and subtype entities

Logical design follows analysis. Logical design converts an entity-relationship model to tables, columns, and keys for a specific database system.

PARTICIPATION ACTIVITY | 3.5.10: Analysis activities.

Match high-level step names to detailed activity descriptions.

If unable to drag and drop, refresh the page.

Determine cardinality **Distinguish strong and weak entities**

Create supertype and subtype entities **Discover entities, relationships, and attributes**

	List standard attribute types in glossary.
	Determine attribute maxima and minima.
	Determine the identifying relationship for each weak entity.
	Identify partitions and partition attributes.

Reset

CHALLENGE ACTIVITY | 3.5.1: Supertype and subtype entities.

544874.3500394.qx3zqy7

Start

Each cooking vessel is made of a particular material, for instance 'metal'. Each pot has a volume measured in quarts. Each pan has a diameter, measured in inches. Each baking dish has a length, width, and height.

Select the correct description for each attribute.

Purpose	Select
Diameter	Select

1

Check   Try again

3.6 Alternative modeling conventions

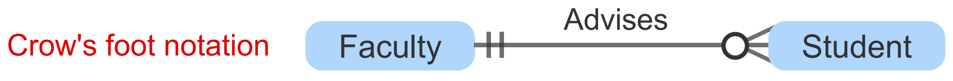
Diagram conventions

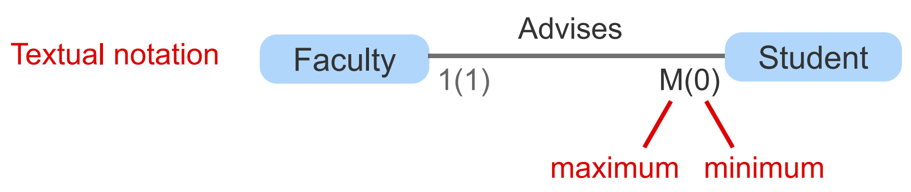
ER diagram conventions vary widely. Ex: Some ER diagrams may:

- Depict relationship names inside a diamond.
- Depict weak entities and identifying relationships with double lines.
- Depict subtype entities with IsA relationships rather than inside of supertype entities.
- Use color, dashed lines, or double lines to convey additional information.

Variations in cardinality symbols are common. Ex: Textual notation depicts optional as 0, singular and required as 1, and plural as M. Maximum cardinality appears first, followed by minimum cardinality in parentheses, as in the animation below.

PARTICIPATION ACTIVITY 3.6.1: Alternative cardinality notation.





Animation content:

Step 1: Each faculty member advises zero or many students. Entities Faculty and Student appear. The relationship Faculty-Advises-Student appears. Along the relationship, crow's foot and circle symbols appear next to Student. The caption crow's foot notation appears.

Step 2. Each student has exactly one faculty advisor. Along the relationship, two short lines appear next to Faculty.

Step 3: In textual notation, M represents maximum many. 0 represents minimum zero. The Faculty-Advises-Student relationship appears a second time, without the crow's foot, circle, or short lines. In the second copy of Faculty-Advises-Student, M(0) appears next to Student. The caption textual notation appears.

Step 4: The maximum always appears first. Minimum follows in parentheses. The caption maximum appears under the M. The caption minimum appears under the 0.

Step 5: 1(1) indicates both minimum and maximum are one. In the second copy of Faculty-Advises-Student, 1(1) appears next to Faculty.

Animation captions:

1. Each faculty member advises zero or many students.
2. Each student has exactly one faculty advisor.
3. In textual notation, M represents maximum many. 0 represents minimum zero.
4. The maximum always appears first. Minimum follows in parentheses.
5. 1(1) indicates both minimum and maximum are one.

PARTICIPATION ACTIVITY | 3.6.2: Alternative diagram conventions.

Match the modeling concept to the alternative diagram notation.

If unable to drag and drop, refresh the page.

plural and optional | identifying relationship | singular and optional | singular and required | plural and required

	1(1)
	M(1)
	Solid double lines
	1(0)
	M(0)

Reset

Model conventions

ER modeling concepts also vary. Ex: Some ER models may:

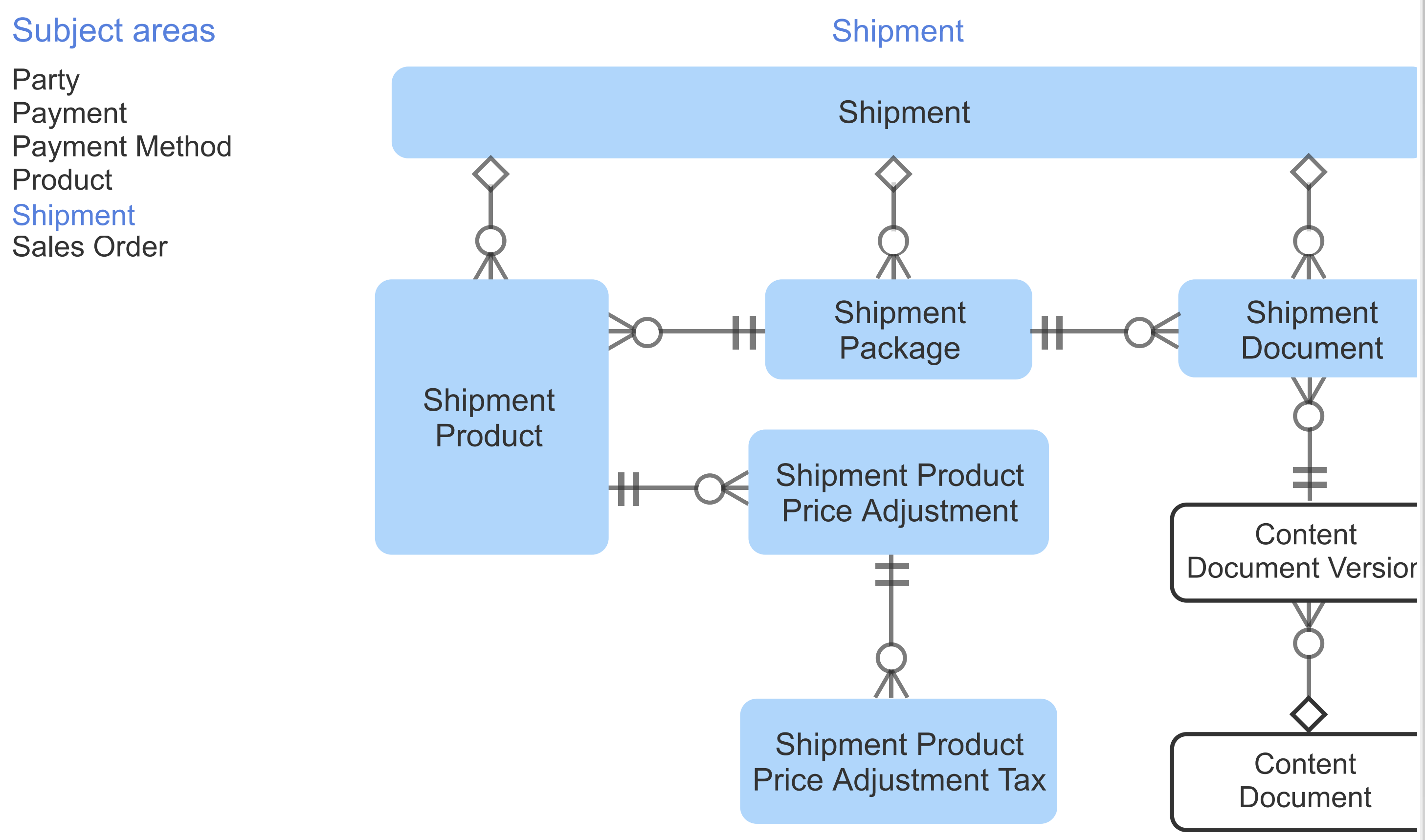
- Allow relationships between three or more entities.
- Decompose a complex model into a group of related entities, called a ***subject area***.
- Refer to strong entities as ***independent*** and weak entities as ***dependent***.

Several model conventions are standardized and widely used. Leading conventions include:

- ***Unified Modeling Language***, or ***UML***, is commonly used for software development. Software data structures are similar to database structures, so UML includes ER conventions.
- ***IDEF1X*** stands for Information DEFinition version 1X. IDEF1X became popular, in part, due to early adoption by the United States Department of Defense.
- ***Chen notation*** appeared in an early ER modeling paper by Peter Chen. Chen notation is not standardized but often appears in literature and tools.

By and large, differences between conventions are stylistic rather than substantial. The choice of convention does not usually affect the resulting database design.

PARTICIPATION ACTIVITY | 3.6.3: Cloud Information Model.



Animation content:

Static figure:
The caption Subject Areas appears. Six subject areas appear under this caption: Party, Payment, Payment Method, Product, Shipment, and Sales Order. Shipment is highlighted.

The caption Shipment appears. Eight entities appear under this caption. Entities Shipment, ShipmentProduct, ShipmentPackage, ShipmentDocument, ShipmentProductPriceAdjustment, and ShipmentProductPriceAdjustmentTax appear with a shaded fill. Entities ContentDocumentVersion and ContentDoucment appear with no fill. The eight entities are connected by unnamed relationships. Cardinality symbols appear in crow's foot notation.

Step 1: The Cloud Information Model is an industry standard for sales fulfillment. The model is divided into subject areas. The subject area names appear.

Step 2: The Shipment subject area contains entities related to product shipments. The Shipment subject area is highlighted. The Shipment caption appears. The entities with shaded fill appear below the caption.

Step 3: Cardinality is depicted with crow's foot notation. Identifying relationships are depicted with diamonds. Relationships and cardinality symbols appear.

Step 4: Related entities outside the Shipment subject area are shown in a different color. The two entities with no fill, and their relationships, appear.

Animation captions:

1. The Cloud Information Model is an industry standard for sales fulfillment. The model is divided into subject areas.
2. The Shipment subject area contains entities related to product shipments.
3. Cardinality is depicted with crow's foot notation. Identifying relationships are depicted with diamonds.
4. Related entities outside the Shipment subject area are shown in a different color.

Source: github.com

PARTICIPATION ACTIVITY 3.6.4: ER model and diagram conventions.

1) In some models, weak entities are called _____ entities.

Check Show answer

2) A group of related entities is often called a/an _____ .

Check Show answer

3) _____ is a modeling standard intended for software development

Check

Show answer