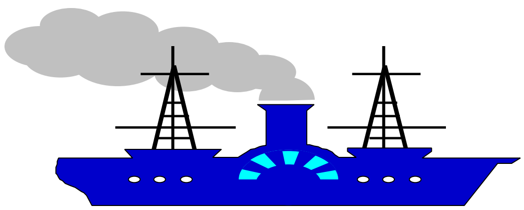
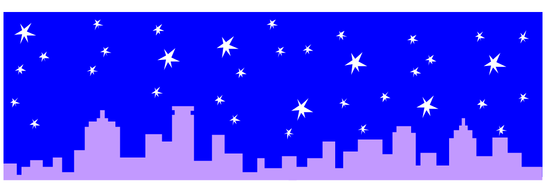
XV. The Entity-Relationship Model

The Entity-Relationship Model
Entities, Relationships and Attributes
Cardinalities, Identifiers and Generalization
Documentation of E-R Diagrams and Business Rules

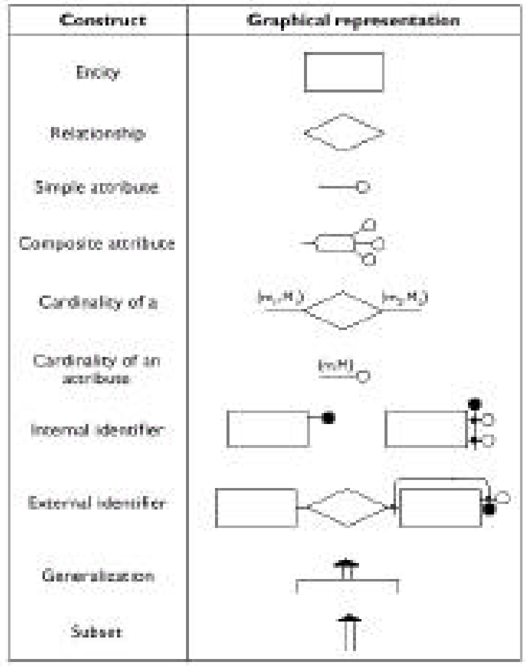


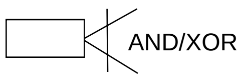
The Entity Relationship Model

- The ***Entity-Relationship*** (ER) ***model*** is a *conceptual* data model, capable of describing the data requirements for a new information system in a direct and easy to understand graphical notation.
- Data requirements are described in terms of a ***conceptual (or, ER) schema***.
- ER schemata are comparable to class diagrams.



The Constructs of the E-R Model

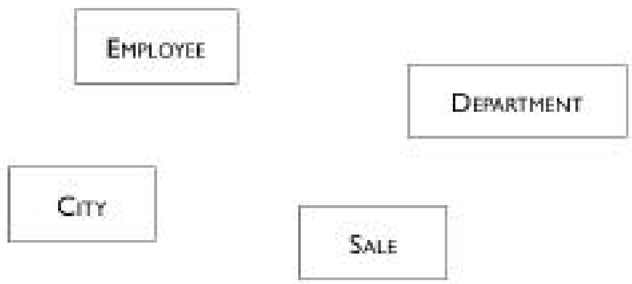




Entities

- These represent classes of objects (facts, things, people,...) that have properties in common and an autonomous existence.
- `City`, `Department`, `Employee`, `Purchase` and `Sale` are examples of entities for a commercial organization.
- An instance of an entity is an object in the class represented by the entity.
- `Stockholm`, `Helsinki`, are examples of instances of the entity `City`, and the employees `Peterson` and `Johanson` are examples of instances of the `Employee` entity.
- The E-R model is very different from the relational model in which it is not possible to represent an object without knowing its properties (an employee is represented by a tuple containing the name, surname, age, and other attributes.)

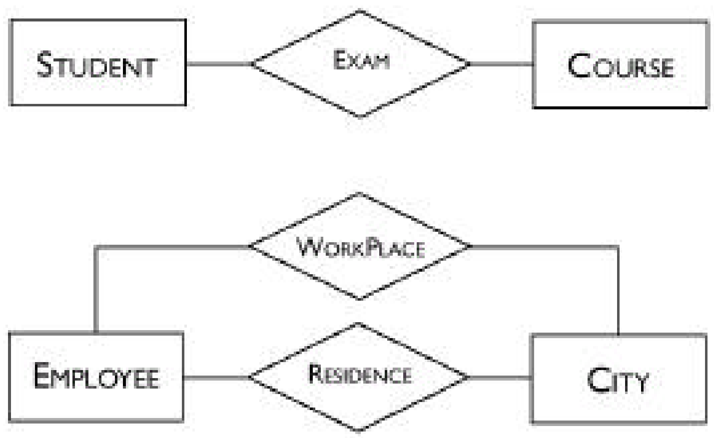
Examples of Entities



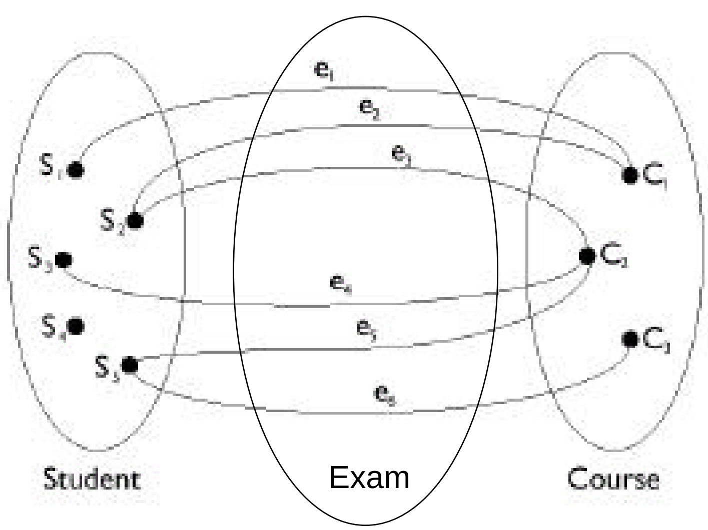
Relationships

- They represent logical links between two or more entities.
- Residence is an example of a relationship that can exist between the entities `City` and `Employee`; `Exam` is an example of a relationship that can exist between the entities `Student` and `Course`.
- An instance of a relationship is an n-tuple made up of instances of entities, one for each of the entities involved.
- The pair of objects made up of the employee named `Johanssen` and the city `Stockholm`, or the pair of objects made from the employee `Peterson` and the city `Oslo`, are examples of instances in the relationship `Residence`.

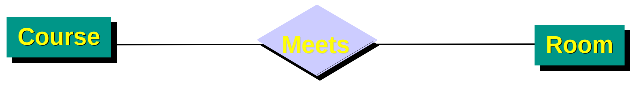
Examples of Relationships



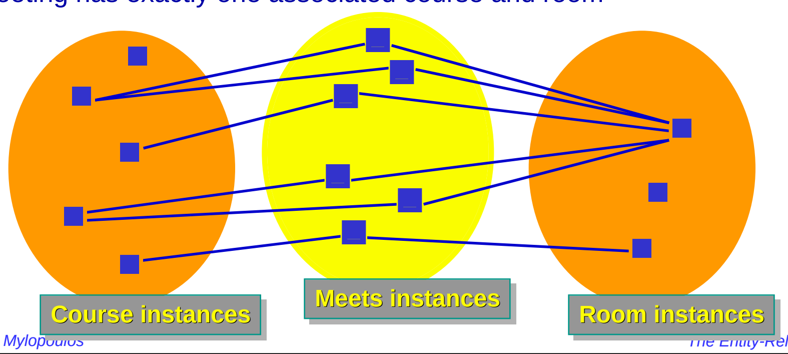
Example of Instances for Exam



What Does An E-R Diagram Really Mean?

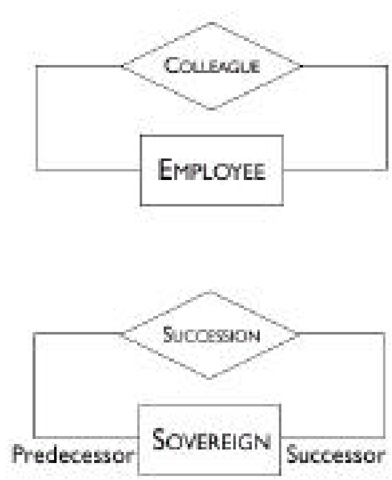


- `Course` and `Room` are entity types. Their instances describe particular courses (e.g., CSC340S) and particular rooms (e.g., NP203).
- `Meets` is a relationship type. Its instances describe particular meetings. Each meeting has exactly one associated course and room

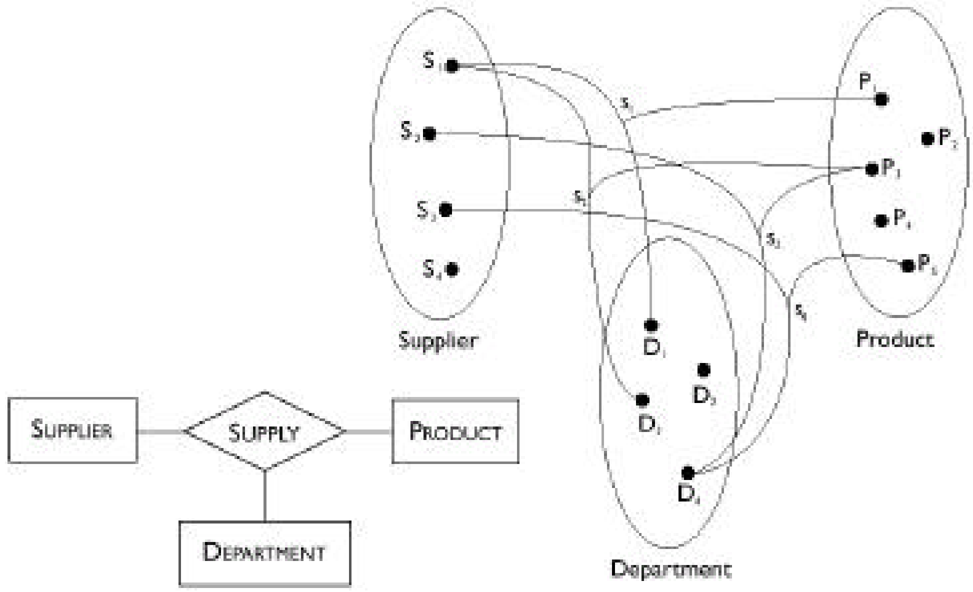


Recursive Relationships

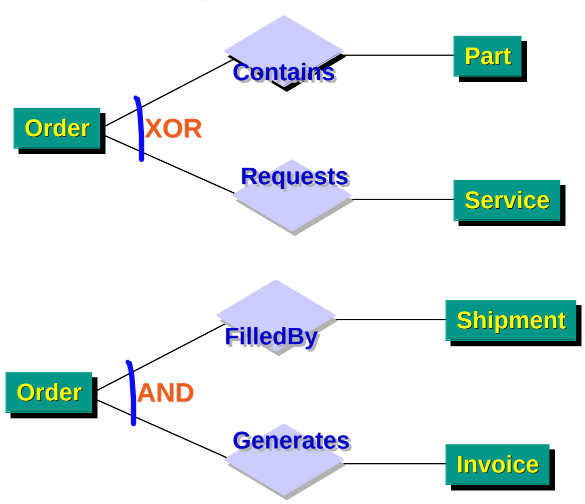
- Recursive relationships are also possible, that is relationships between an entity and itself.
- Note in the second example that the the relationship is not symmetrical. In this case it is necessary to indicate the two roles that the entity involved plays in the relationship.



Ternary Relationships

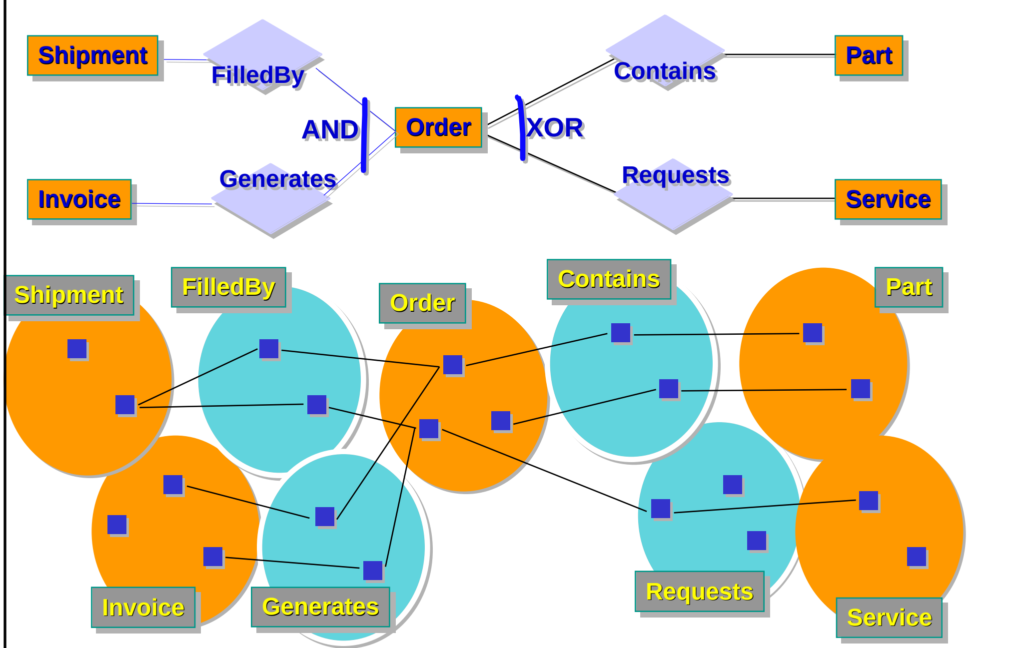


AND/XOR for Relationships



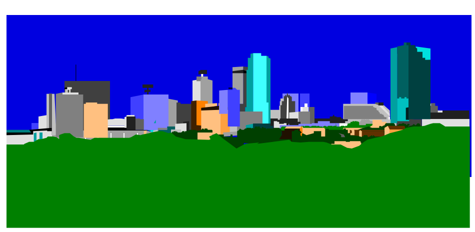
"Orders either order a part or request a service, but not both"

"For any given order, whenever there is at least one invoice there is also at least one shipment and vice versa"

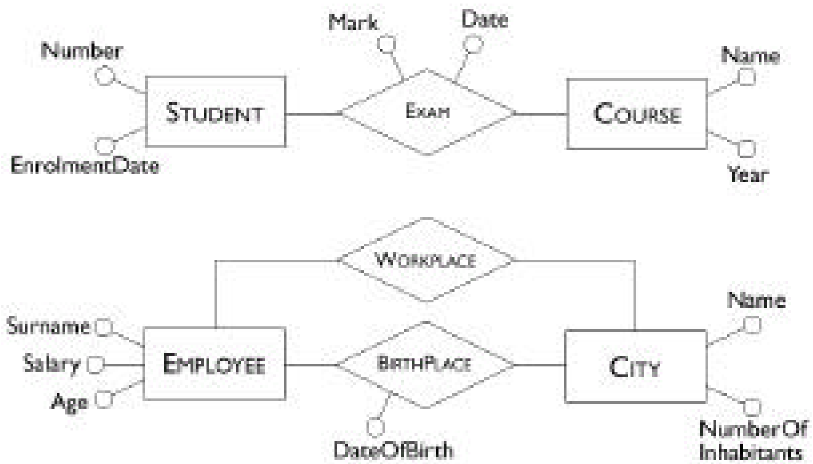


Attributes

- These describe the elementary properties of entities or relationships.
- For example, `Surname`, `Salary` and `Age` are possible attributes of the `Employee` entity, while `Date` and `Mark` are possible attributes for the relationship `Exam` between `Student` and `Course`.
- An attribute associates with each instance of an entity (or relationship) a value belonging to a set known as the ***domain*** of the attribute.
- The domain contains the admissible values for the attribute.

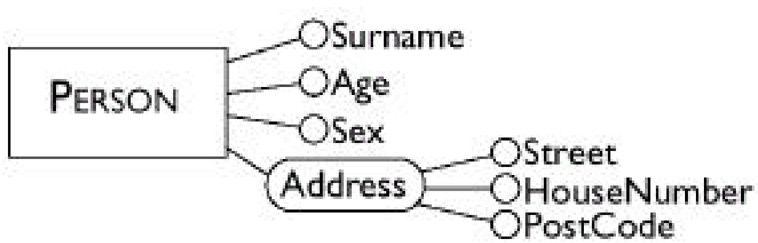


Attribute Examples

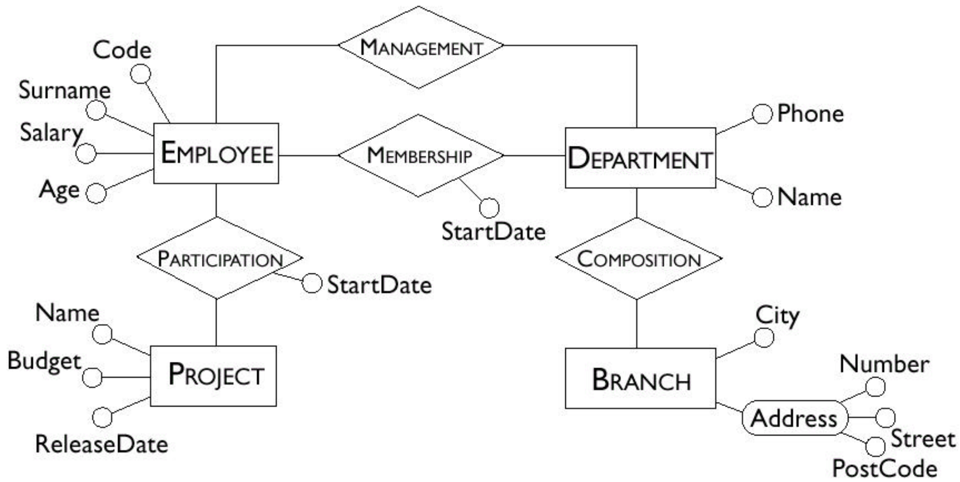


Composite Attributes

- It is sometimes convenient to group attributes of the same entity or relationship that have closely connected meanings or uses. Such groupings are called ***composite attributes***.



Schema with Attributes



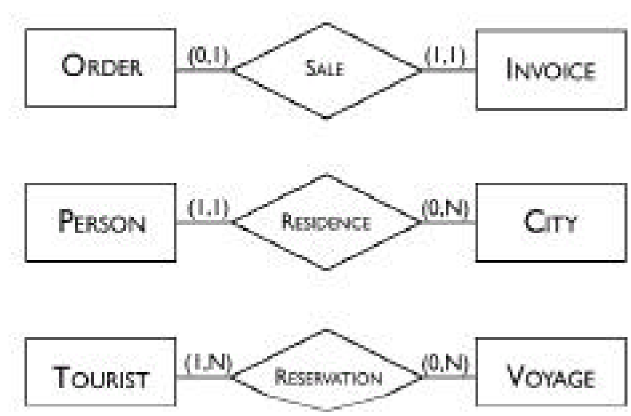
Cardinalities

- These are specified for each entity participating in a relationship and describe the maximum and minimum number of relationship occurrences in which an entity occurrence can participate.
- Cardinalities state how many times can an entity instance participate in instances of a given relationship.

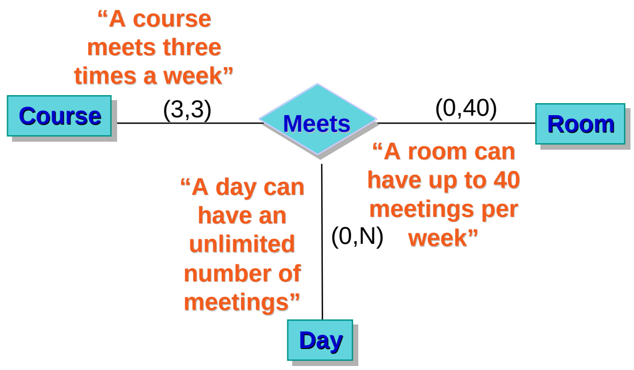
Cardinalities (cont'd)

- In principle, a cardinality is any pair of non-negative integers (n,m) such that n≤m. or a pair of the form (n,N) where N means "any number".
- If minimum cardinality is 0, we say that entity participation in a relationship is optional. If minimum cardinality is 1, we say that entity participation in a relationship is mandatory.
- If maximum cardinality is 1, each instance of the entity is associated at most with a single instance of the relationship; if maximum cardinality is N, then each instance of the entity is associated with an arbitrary number of instances of the relationship.

Cardinality Examples

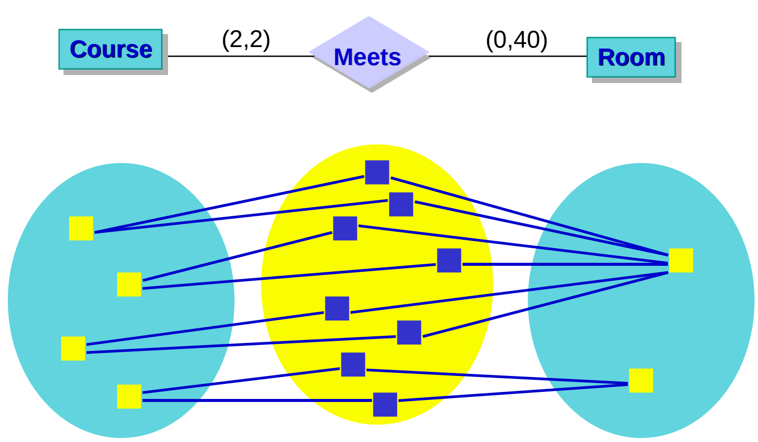


Cardinality Example

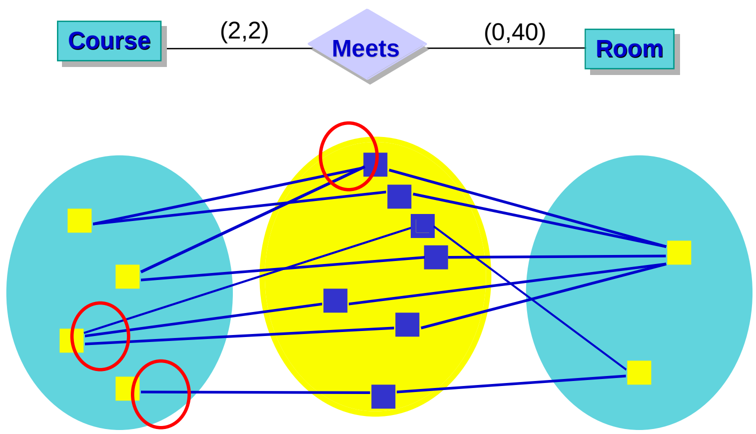


Instantiating ER Diagrams

- An ER diagram specifies what states are possible in the world being modeled

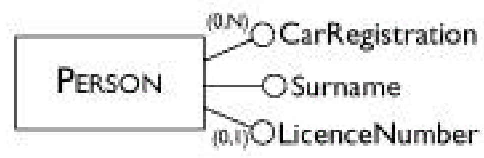


Illegal Instantiations



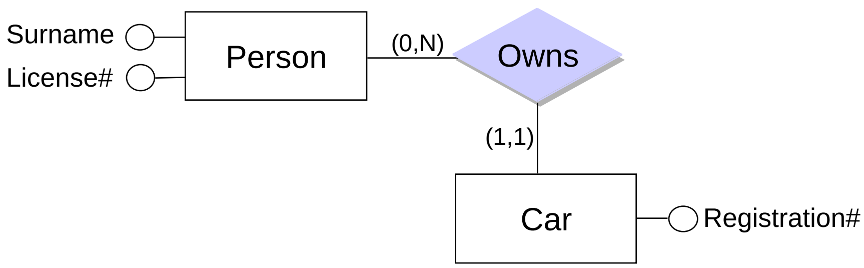
Cardinalities of Attributes

- They are specified for the attributes of entities (or relationships) and describe the minimum and maximum number of values of the attribute associated with instances of an entity or a relationship.
- In most cases, the cardinality of an attribute is equal to (1,1) and is omitted (***single-valued*** attributes)
- The value of a certain attribute however, may also be null, or there may exist several values of a certain attribute for an entity instance (***multi-valued*** attributes)



Cardinalities (cont'd)

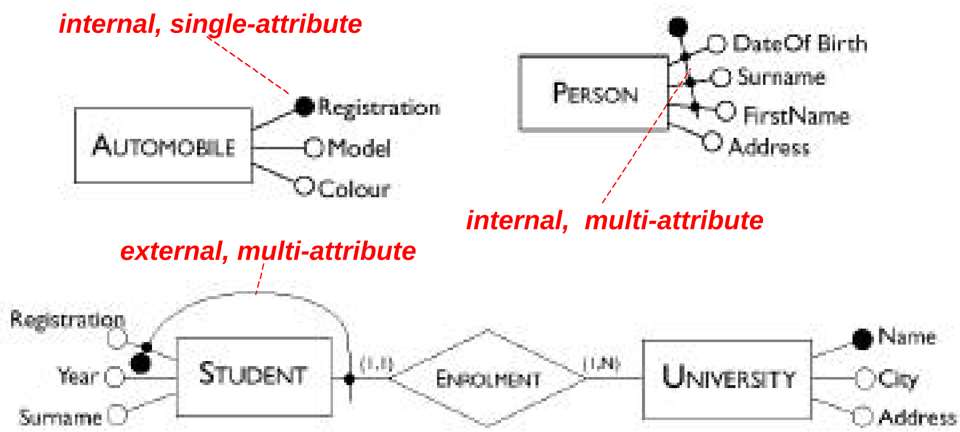
- Multi-valued attributes should be used with great caution, because they represent situations that can be modelled in many cases with additional entities linked by one-to-many (or many-to-many) relationships to the entity to which they refer.



Identifiers

- ***Identifiers*** (or ***keys***) consist of one or more attributes which identify uniquely instances of an entity.
- In many cases, an identifier is formed by one or more attributes of the entity itself: in this case we talk about an ***internal*** identifier.
- Sometimes, however, the attributes of an entity are not sufficient to identify its instances unambiguously and other entities are involved in the identification. Identifiers of this type are called ***external*** identifiers.
- An identifier for a relationship consists of identifiers for all the entities it relates. For example, the identifier for the relationship (Person-) Owns(-Car) is a combination of the Person and Car identifiers.

Examples of Identifiers



General Observations on Identifiers

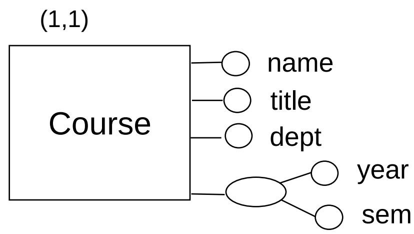
- An identifier can involve one or more attributes, provided that each of them has (1,1) cardinality;
- An external identifier can involve one or more entities, provided that each of them is member of a relationship to which the entity to identify participates with cardinality equal to (1,1);
- An external identifier can involve an entity that is in its turn identified externally, as long as cycles are not generated;
- Each entity must have one (internal or external) identifier, but can have more than one. Actually, if there is more than one identifier, then the attributes and entities involved in an identification can be optional (minimum cardinality equal to 0).

Modeling an Application with Identifiers

Identifiers constitute a powerful mechanism for modeling an application. Assume we want a database storing information about lecture meetings.

- Suppose first that we use the identifier `coursename,day,hour` for the `Meeting` entity. This says, that there can only be one meeting at any one time for a given course name, day, hour; in other words, we can't have two sections of the same course meeting at the same day+hour.
- Suppose now we use only `coursename` as identifier for `Meeting`. This says that there can only be one meeting per given course name (unreasonable!)
- If we use `courseinstructor,room` as identifier for `Meeting`, we are stating that there can only be one meeting for a given instructor+room combination, so an instructor must have all her meetings in different rooms!
- Finally, if the attribute `courseinstructor` by itself forms an identifier for `Meeting`, then the diagram we have built is stating that each instructor participates in at most one meeting, again this is unreasonable.

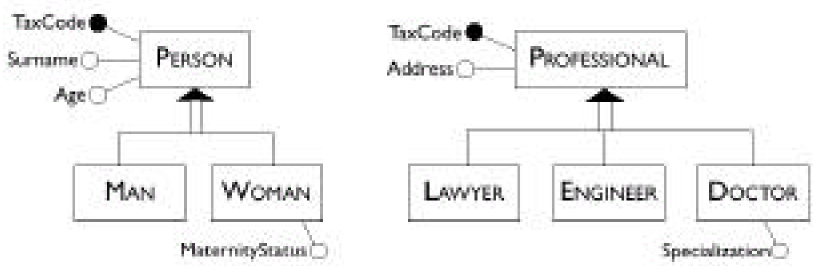
Consider...



- Assume that we want to keep track of course offerings over the years.
- What does each of the following combinations of attributes say about the application?
  - ✓name;
  - ✓name, sem, year;
  - ✓sem, year;
  - ✓name, dept;
  - ✓dept, year.

Generalizations

- These represent logical links between an entity E, known as ***parent*** entity, and one or more entities $E_1,...,E_n$ called ***child*** entities, of which E is more general, in the sense that they are a particular case.
- In this situation we say that E is a ***generalization*** of $E_1,...,E_n$ and that the entities $E_1,...,E_n$ are ***specializations*** of E.



Properties of Generalization

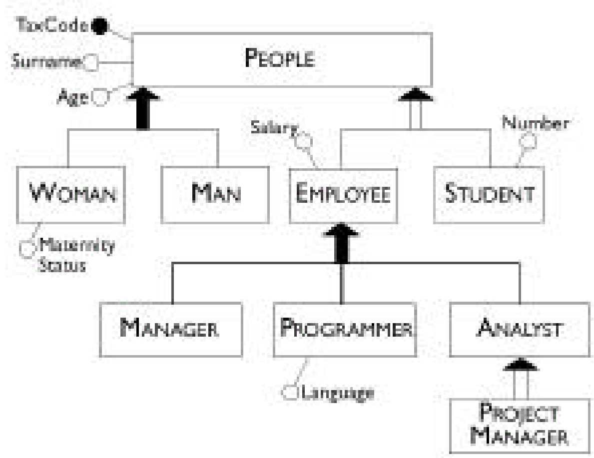
- Every instance of a child entity is also an instance of the parent entity.
- Every property of the parent entity (attribute, identifier, relationship or other generalization) is also a property of a child entity. This property of generalizations is known as ***inheritance***.

Types of Generalizations

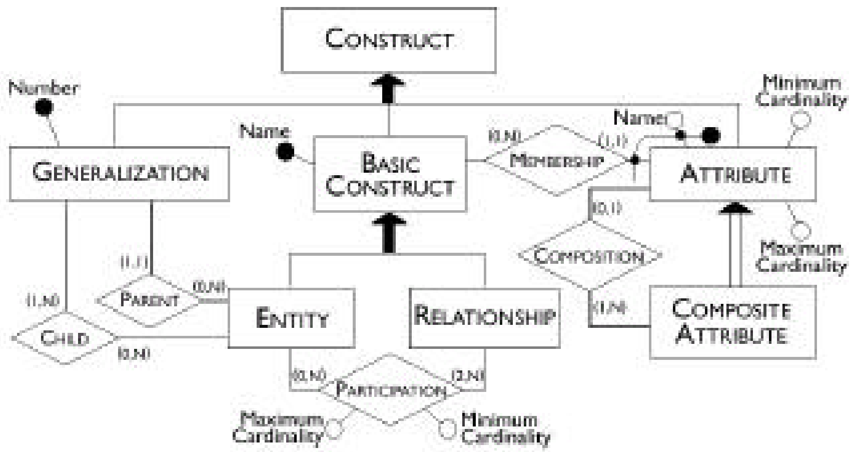
- A generalization is ***total*** if every instance of the parent entity is also an instance of one of its children, otherwise it is ***partial***.
- A generalization is ***exclusive*** if every instance of the parent entity is at most an instance of one of the children, otherwise it is ***overlapping***.
- The generalization `Person`, of `Man` and `Woman` is total (the sets of men and the women constitute 'all' the people) and exclusive (a person is either a man or a woman).
- The generalization `Vehicle` of `Automobile` and `Bicycle` is partial and exclusive, because there are other types of vehicle (for example, motor bike) that are neither cars nor bicycle.
- The generalization `Person` of `Student` and `Employee` is partial and overlapping, because there are students who are also employed.

Generalization Hierarchies

- Total generalization is represented by a solid arrow.
- In most applications, modeling the domain involves a *hierarchy* of generalizations that includes several levels



The E-R Model, as an E-R Diagram



Documentation of an E-R Schema

- An Entity-Relationship schema is rarely sufficient by itself to represent all the aspects of an application in detail.
- It is therefore important to complement every E-R schema with support documentation, which can facilitate the interpretation of the schema itself and describe properties of the data that cannot be expressed directly by the constructs of the model.
- A widely-used documentation concept for conceptual schemas is the ***business rule***.

Business Rules

- Business rules are used to describe the properties of an application, e.g., the fact that an employee cannot earn more than his or her manager.
- A business rule can be:
  - ✓ the ***description*** of a concept relevant to the application (also known as a ***business object***),
  - ✓ an ***integrity constraint*** on the data of the application,
  - ✓ a ***derivation rule***, whereby information can be derived from other information within a schema.

Documentation Techniques

- Descriptive business rules can be organized as a *data dictionary*. This is made up of two tables: the first describes the entities of the schema, the others describes the relationships.
- Business rules that describe constraints can be expressed in the following form:

  <concept> must/must not <expression on concepts>
- Business rules that describe derivations can be expressed in the following form:

  <concept> is obtained by <operations on concepts>

Example of a Data Dictionary

Entity	Description	Attributes	Identifier
EMPLOYEE	Employee working in the company.	Code, Surname, Salary, Age	Code
PROJECT	Company project on which employees are working.	Name, Budget, ReleaseDate	Name
....	....	....	....

Relationship	Description	Entities involved	Attributes
MANAGEMENT	Associate a manager with a department.	Employee (0,1), Department (1,1)	
MEMBERSHIP	Associate an employee with a department.	Employee (0,1) Department (1,N)	StartDate
....	....	....	....

Examples of Business Rules

Constraints
(BR1) The manager of a department must belong to that department. **(BR2)** An employee must not have a salary greater than that of the manager of the department to which he or she belongs. **(BR3)** A department of the Rome branch must be managed by an employee with more than 10 years' employment with the company. **(BR4)** An employee who does not belong to a particular department must not participate in any project. ....
Derivations
(BR5) The budget for a project is obtained by multiplying the sum of the salaries of the employees who are working on it by 3. ....

Comparison of ER and Class Diagrams

- ER diagrams allow N-ary relationships, N≥2; Class diagrams only allow binary relationships.
- ER diagrams allow multi-valued attributes, class diagrams do not.
- ER diagrams allow the specification of identifiers (an often-encountered type of constraint), while class diagrams do not.
- Class diagrams allow dynamic classification, but ER diagrams do not.