CO226: Database Systems

EER Modeling

Sampath Deegalla
dsdeegalla@pdn.ac.lk

10th July 2014

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Enhanced-ER (EER) Model Concepts

- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Subclasses and Superclasses

- An entity type may have additional meaningful subgroupings of its entities
    - Example: EMPLOYEE may be further grouped into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE, ...
- These are called superclass/subclass relationships.
    - Example: EMPLOYEE/SECRETARY, EMPLOYEE/TECHNICIAN

Subclasses and Superclasses

- An entity type may have additional meaningful subgroupings of its entities
  - Example: EMPLOYEE may be further grouped into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE, . . .
- These are called superclass/subclass relationships.
  - Example: EMPLOYEE/SECRETARY, EMPLOYEE/TECHNICIAN

Subclasses and Superclasses

- An entity type may have additional meaningful subgroupings of its entities
  - Example: EMPLOYEE may be further grouped into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE, . . .
- These are called superclass/subclass relationships.
  - Example: EMPLOYEE/SECRETARY, EMPLOYEE/TECHNICIAN

Subclasses and Superclasses

- These are also called IS-A relationships (SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, . . .).
- Note: An entity that is member of a subclass represents the same real-world entity as some member of the superclass
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED_EMPLOYEE
  - It is not necessary that every entity in a superclass be a member of some subclass

Subclasses and Superclasses

- These are also called IS-A relationships (SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, . . .).
- Note: An entity that is member of a subclass represents the same real-world entity as some member of the superclass
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED_EMPLOYEE
    - It is not necessary that every entity in a superclass be a member of some subclass

Subclasses and Superclasses

- These are also called IS-A relationships (SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, . . .).
- Note: An entity that is member of a subclass represents the same real-world entity as some member of the superclass
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED_EMPLOYEE
  - It is not necessary that every entity in a superclass be a member of some subclass

Subclasses and Superclasses

- These are also called IS-A relationships (SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, . . .).
- Note: An entity that is member of a subclass represents the same real-world entity as some member of the superclass
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED_EMPLOYEE
  - It is not necessary that every entity in a superclass be a member of some subclass

Attribute Inheritance in Superclass/ Subclass Relationships

- An entity that is member of a subclass inherits all attributes of the entity as a member of the superclass
- It also inherits all relationships

Attribute Inheritance in Superclass/ Subclass Relationships

- An entity that is member of a subclass inherits all attributes of the entity as a member of the superclass
- It also inherits all relationships

Attribute Inheritance in Superclass/ Subclass Relationships

- An entity that is member of a subclass inherits all attributes of the entity as a member of the superclass
- It also inherits all relationships

Specialization

- Is the process of defining a set of subclasses of a superclass
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass
- Example: SECRETARY, ENGINEER, TECHNICIAN is a specialization of EMPLOYEE based upon job type.
- Example: Another specialization of EMPLOYEE based in method of pay is SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.

Specialization

- Is the process of defining a set of subclasses of a superclass
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass
- Example: SECRETARY, ENGINEER, TECHNICIAN is a specialization of EMPLOYEE based upon job type.
- Example: Another specialization of EMPLOYEE based in method of pay is SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.

Specialization

- Is the process of defining a set of subclasses of a superclass
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass
- Example: SECRETARY, ENGINEER, TECHNICIAN is a specialization of EMPLOYEE based upon job type.
- Example: Another specialization of EMPLOYEE based in method of pay is SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.

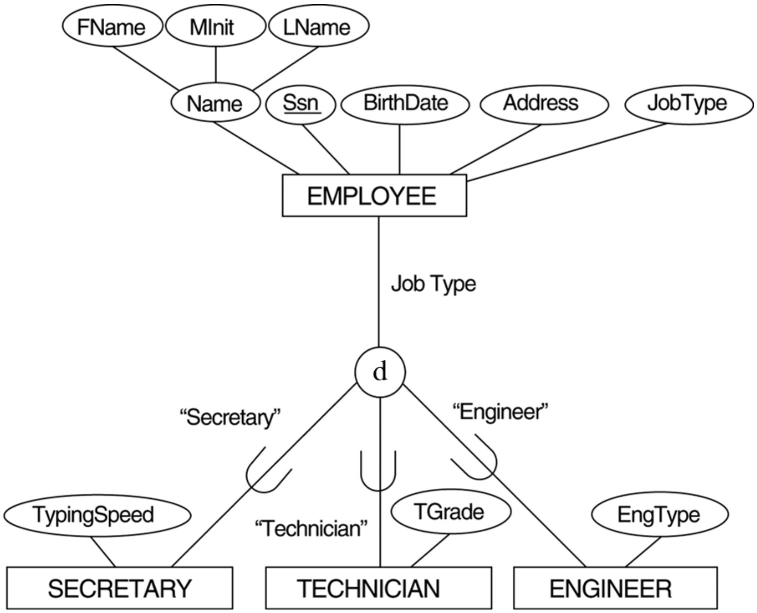
Specialization

- Is the process of defining a set of subclasses of a superclass
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass
- Example: SECRETARY, ENGINEER, TECHNICIAN is a specialization of EMPLOYEE based upon job type.
- Example: Another specialization of EMPLOYEE based in method of pay is SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.

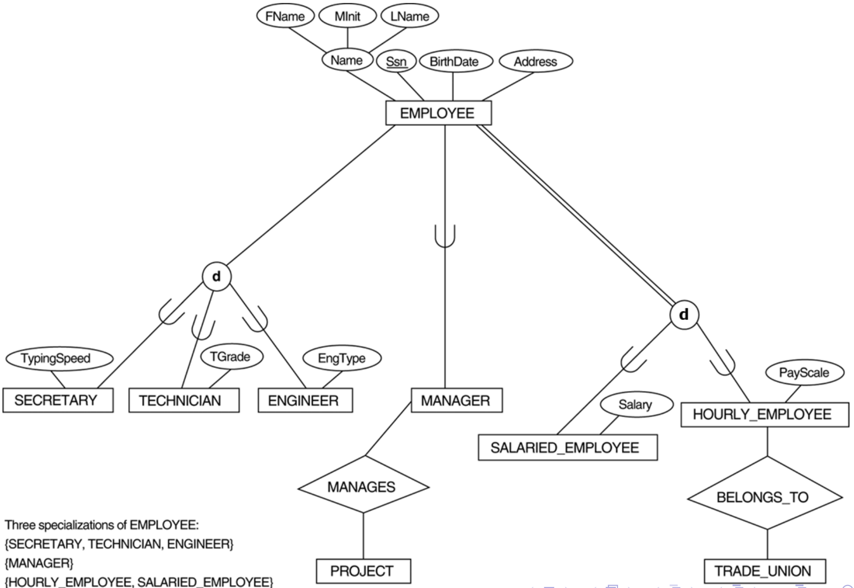
Specialization

- Is the process of defining a set of subclasses of a superclass
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass
- Example: SECRETARY, ENGINEER, TECHNICIAN is a specialization of EMPLOYEE based upon job type.
- Example: Another specialization of EMPLOYEE based in method of pay is SALARIED_EMPLOYEE, HOURLY_EMPLOYEE.

Example of a Specialization



EER diagram notation to represent subclasses and specialization



Three specializations of EMPLOYEE:
{SECRETARY, TECHNICIAN, ENGINEER}
{MANAGER}
{HOURLY_EMPLOYEE, SALARIED_EMPLOYEE}

Generalization

- The reverse of the specialization process
- Several classes with common features are generalized into a superclass; original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE; both CAR, TRUCK become subclasses of the superclass VEHICLE.
  - We can view CAR, TRUCK as a specialization of VEHICLE
  - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

Generalization

- The reverse of the specialization process
- Several classes with common features are generalized into a superclass; original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE; both CAR, TRUCK become subclasses of the superclass VEHICLE.
  - We can view CAR, TRUCK as a specialization of VEHICLE
  - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

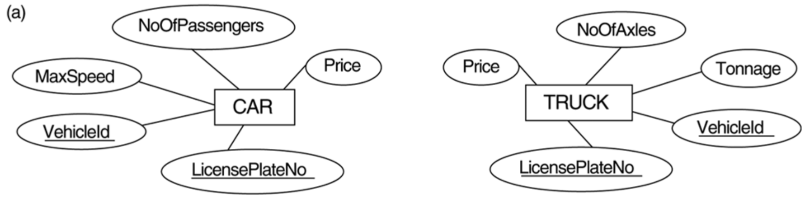
Generalization

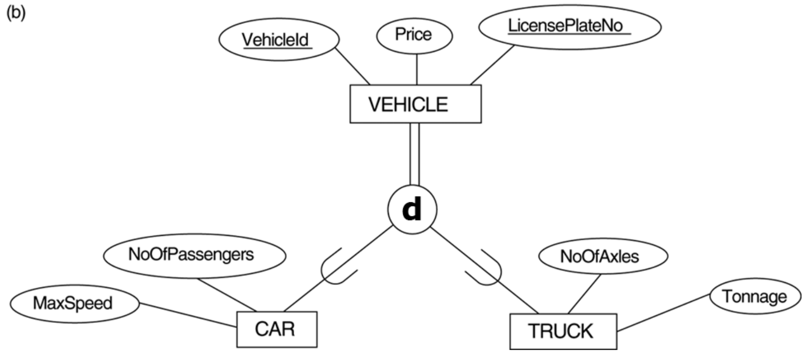
- The reverse of the specialization process
- Several classes with common features are generalized into a superclass; original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE; both CAR, TRUCK become subclasses of the superclass VEHICLE.
    - We can view CAR, TRUCK as a specialization of VEHICLE
    - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

Generalization

- The reverse of the specialization process
- Several classes with common features are generalized into a superclass; original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE; both CAR, TRUCK become subclasses of the superclass VEHICLE.
    - We can view CAR, TRUCK as a specialization of VEHICLE
    - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

Examples of generalization





Constraints on Specialization/Generalization

- Disjointness Constraint
  - Specifies that the subclasses of the specialization must be disjointed (an entity can be a member of at most one of the subclasses of the specialization)
  - Specified by d in EER diagram
  - If not disjointed, overlap; that is the same entity may be a member of more than one subclass of the specialization
  - Specified by o in EER diagram
- Completeness Constraint
  - Total specifies that every entity in the superclass must be a member of some subclass in the specialization/ generalization
  - Shown in EER diagrams by a double line
  - Partial allows an entity not to belong to any of the subclasses
  - Shown in EER diagrams by a single line

Constraints on Specialization/Generalization

- Disjointness Constraint
  - Specifies that the subclasses of the specialization must be disjointed (an entity can be a member of at most one of the subclasses of the specialization)
  - Specified by d in EER diagram
  - If not disjointed, overlap; that is the same entity may be a member of more than one subclass of the specialization
  - Specified by o in EER diagram
- Completeness Constraint
  - Total specifies that every entity in the superclass must be a member of some subclass in the specialization/ generalization
  - Shown in EER diagrams by a double line
  - Partial allows an entity not to belong to any of the subclasses
  - Shown in EER diagrams by a single line

Constraints on Specialization/Generalization

- Disjointness Constraint
  - Specifies that the subclasses of the specialization must be disjointed (an entity can be a member of at most one of the subclasses of the specialization)
  - Specified by d in EER diagram
  - If not disjointed, overlap; that is the same entity may be a member of more than one subclass of the specialization
  - Specified by o in EER diagram
- Completeness Constraint
  - Total specifies that every entity in the superclass must be a member of some subclass in the specialization/ generalization
  - Shown in EER diagrams by a double line
  - Partial allows an entity not to belong to any of the subclasses
  - Shown in EER diagrams by a single line

Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
    - Disjoint, total
    - Disjoint, partial
    - Overlapping, total
    - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

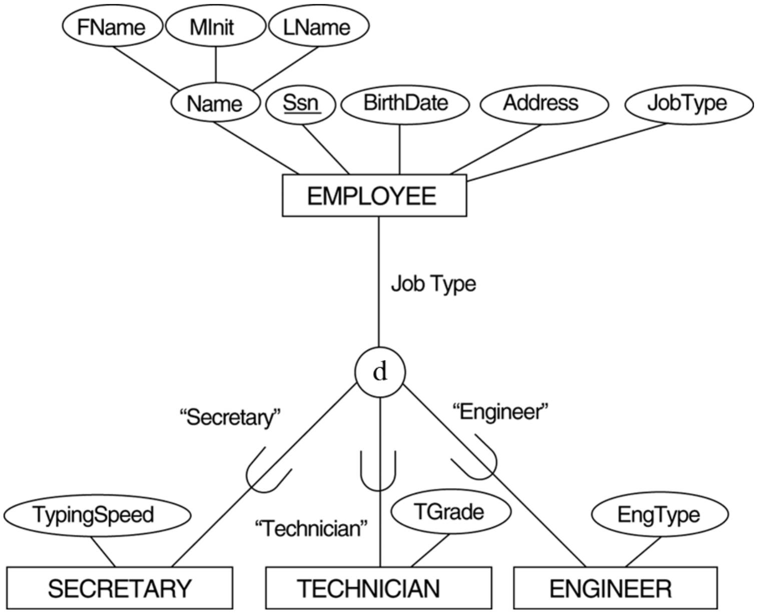
Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

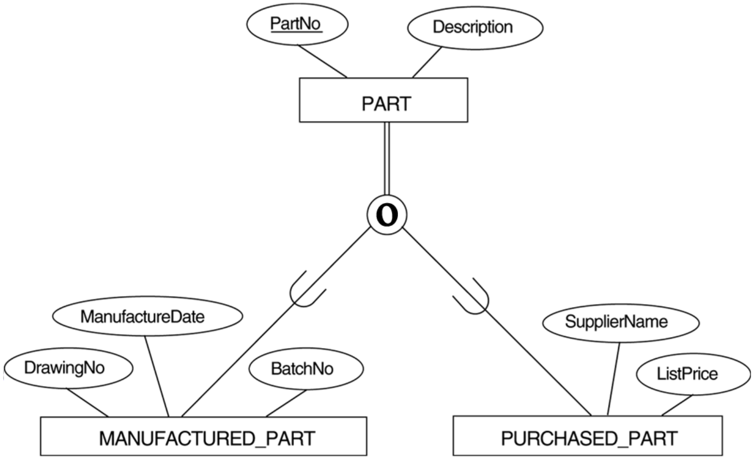
Constraints on Specialization/ Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Example of disjoint partial specialization



Example of overlapping specialization



Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

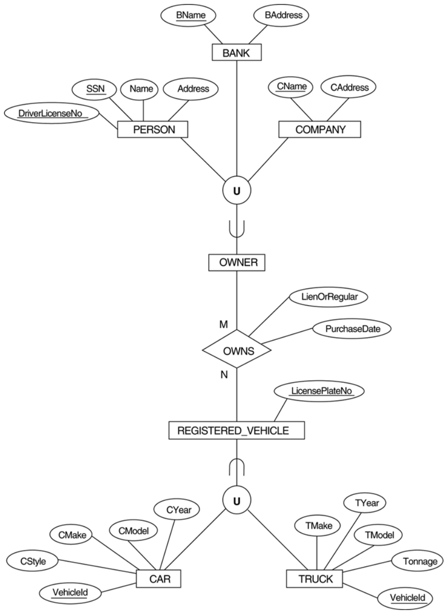
Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass
- A shared subclass is subclass in more than one distinct superclass/subclass relationships, where each relationships has a single superclass (multiple inheritance)
- In some cases, need to model a single superclass/subclass relationship with more than one superclass
- Superclasses represent different entity types
- Such a subclass is called a category or UNION TYPE

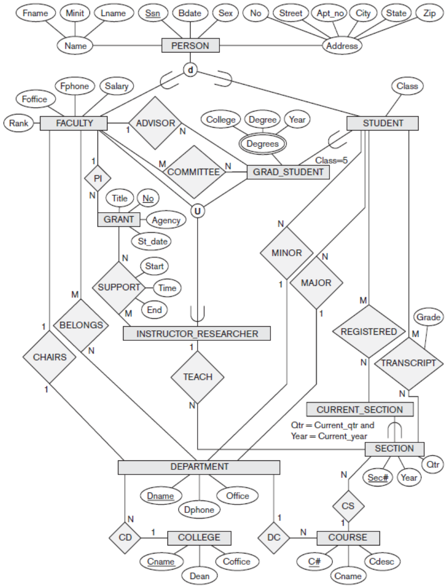
Categories (UNION TYPES)

- Example: Database for vehicle registration, vehicle owner can be a person, a bank (holding a lien on a vehicle) or a company.
    - Category (subclass) OWNER is a subset of the union of the three superclasses COMPANY, BANK, and PERSON
    - A category member must exist in at least one of its superclasses

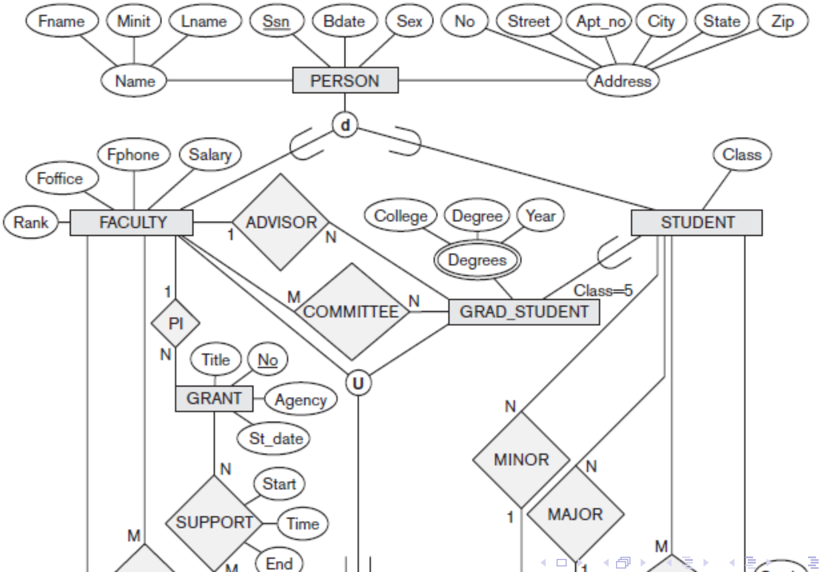
Example of categories (UNION TYPES)



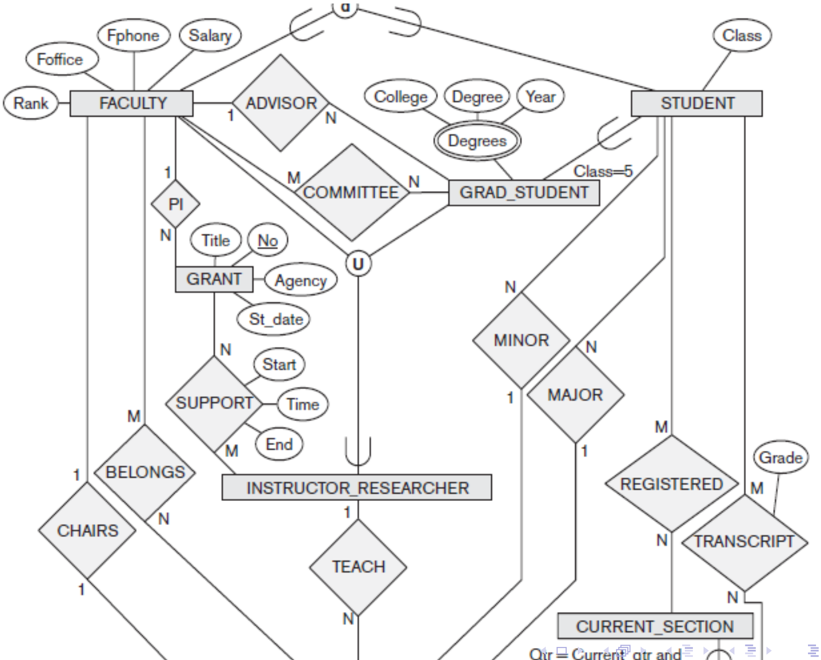
An EER conceptual schema for a UNIVERSITY database.



An EER conceptual schema for a UNIVERSITY database.



An EER conceptual schema for a UNIVERSITY database.


d
Fphone
Salary
Foffice
Rank
FACULTY
ADVISOR
1
N
Class
STUDENT
College
Degree
Year
Degrees
M
COMMITTEE
N
GRAD_STUDENT
Class=5
1
PI
N
Title
No
GRANT
Agency
St_date
U
N
MINOR
N
MAJOR
1
1
N
SUPPORT
Start
Time
End
M
M
BELONGS
1
CHAIRS
N
1
INSTRUCTOR_RESEARCHER
1
TEACH
N
M
REGISTERED
N
M
Grade
TRANSCRIPT
N
CURRENT_SECTION
Qtr = Current_qtr and