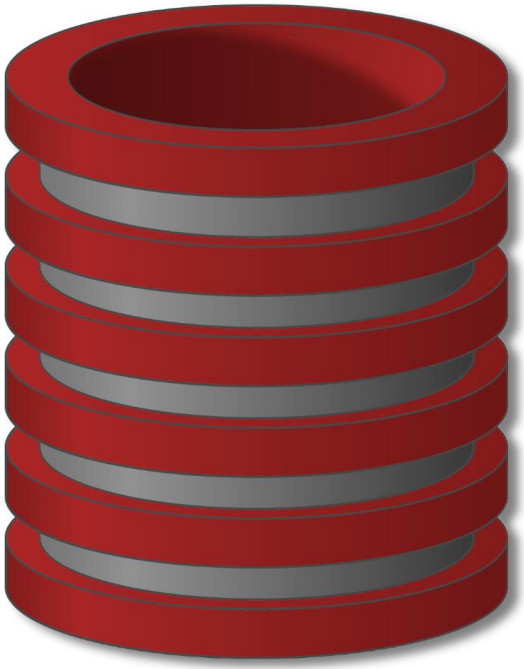


UML

UML Data Modeling

참고자료

UML (Unified Modeling Language)

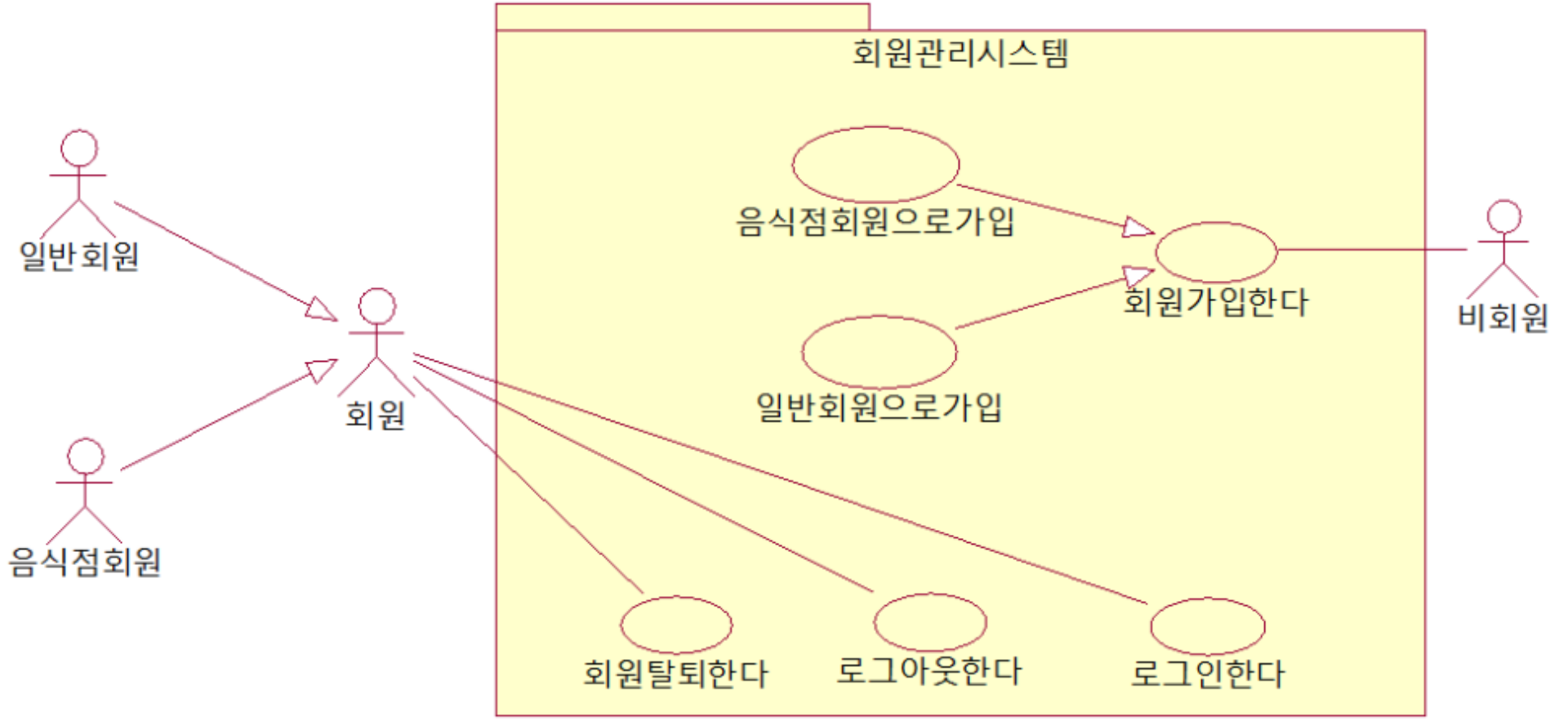
- Standard by OMG (Object Management Group)
  - UML : modeling language in the field of S/W engineering, that is intended to provide a standard way to visualize the design of a system.

- UML has many components (10 Diagrams) to graphically model different aspects of an entire software system
  - Behavior Diagrams : Use case / Activity Diagram
  - Interaction Diagrams : Communication / Sequence Diagram
  - Structure Diagrams : Class Diagram / Component Diagram
  - …….

- Various Application besides Database Modeling
  - Software design
  - Hardware design

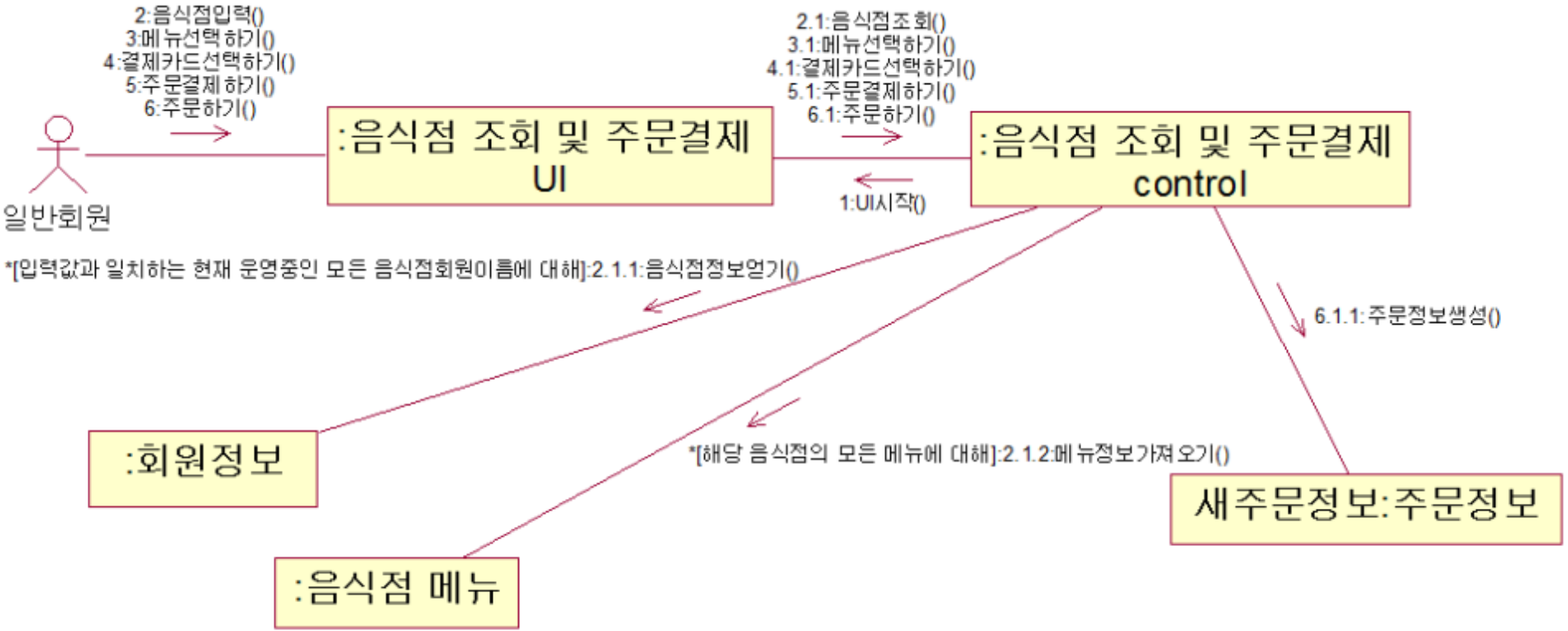
UML Example : Use case diagram

참고자료



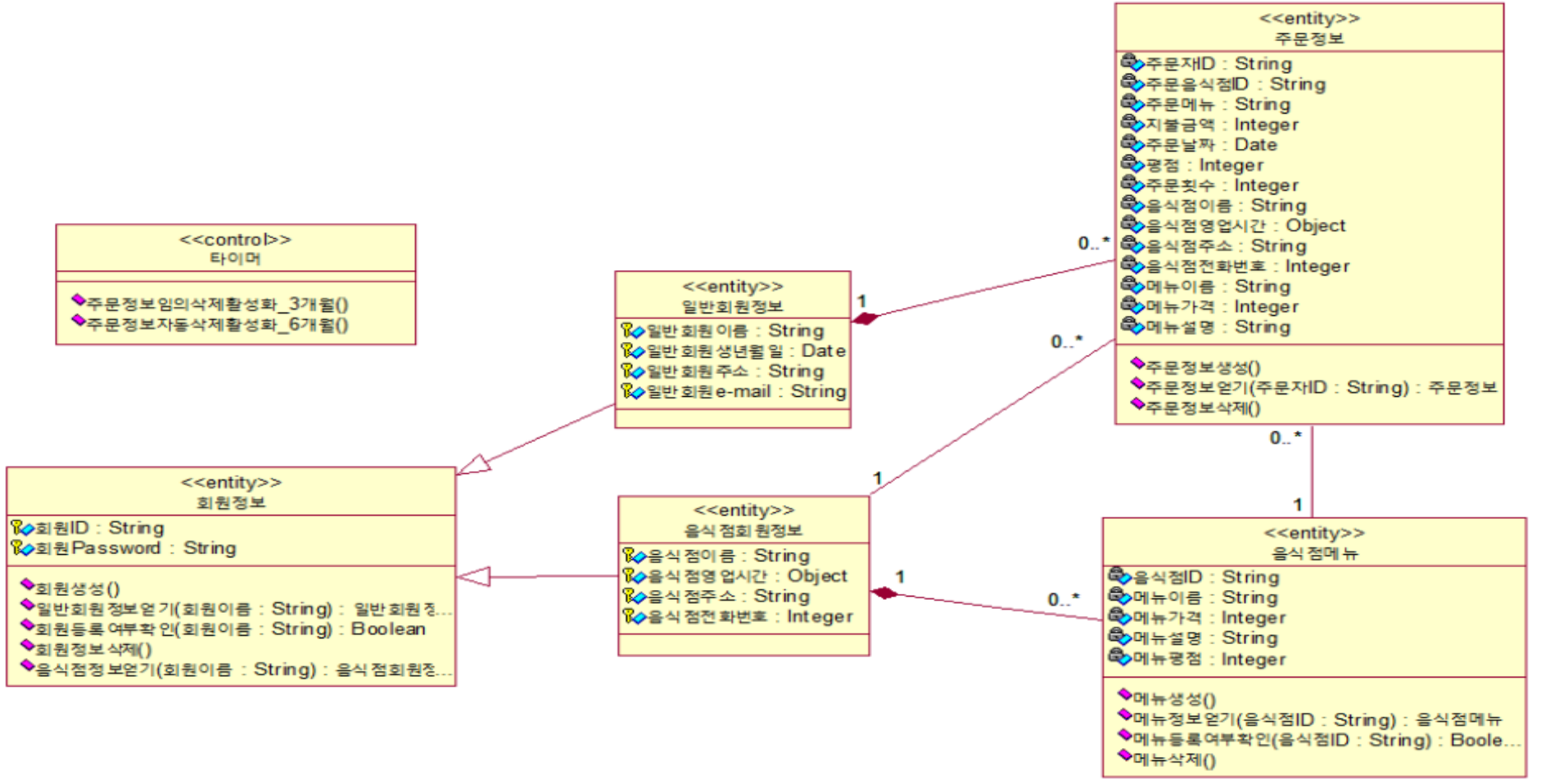
UML Example : Communication diagram

참고자료



UML Example : Class diagram

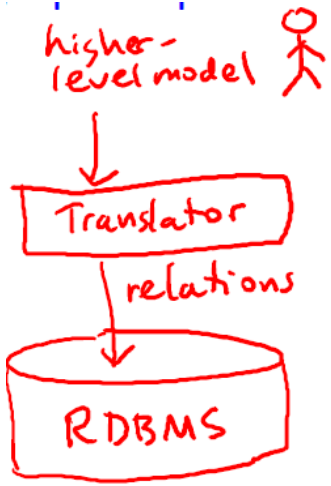
참고자료



Data Modeling

How to represent data for application

- Relational model – with design principles
- XML
- Database design model
  - Not implemented by system
  - Translated into model of DBMS



Higher-Level Database Design Models

- *Entity-Relationship Model* (E/R)
- *Unified Modeling Language* (UML)
  - Data modeling subset
- Both are graphical
- Both can be translated to relations automatically
  - Or semi-automatically

UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

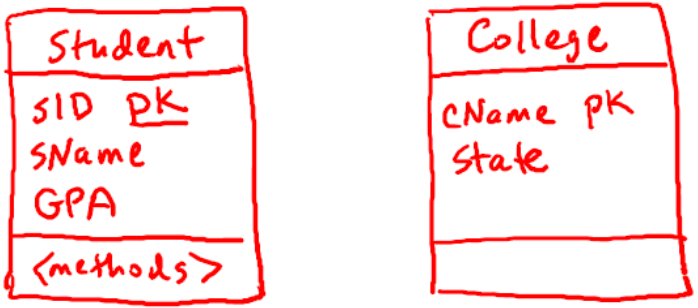
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

UML Data Modeling: **Classes**

Name, attributes, methods



UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

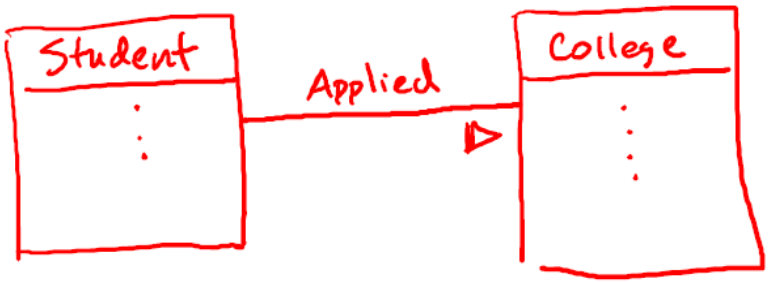
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

UML Data Modeling: **Associations**

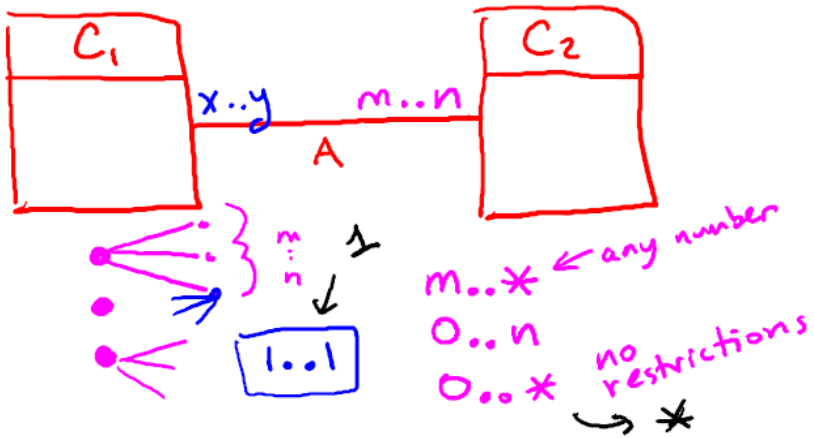
Relationships between objects of two classes



Multiplicity of Associations

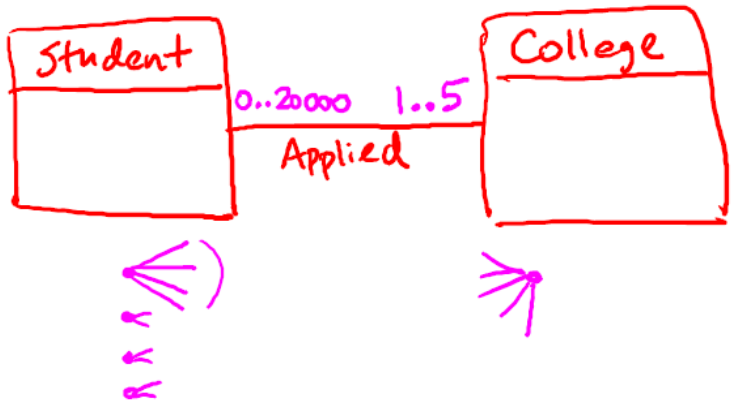
Relationships between objects of two classes

Each object of class C_1 is related to at least m and at most n objects of class C_2



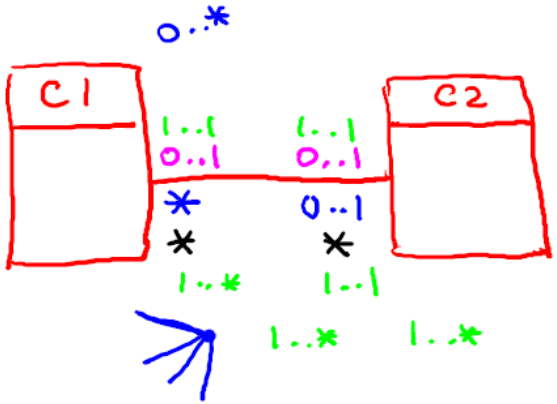
Multiplicity of Associations: Example

Students must apply somewhere and may not apply to more than 5 colleges. No college takes more than 20,000 applications.



Multiplicity of Associations: Types of Relationships

- One-to-One
- Many-to-One
- Many-to-Many
- Complete



UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

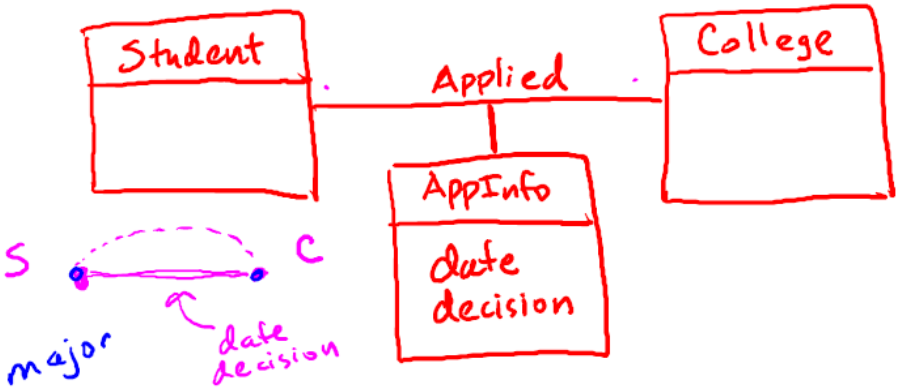
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

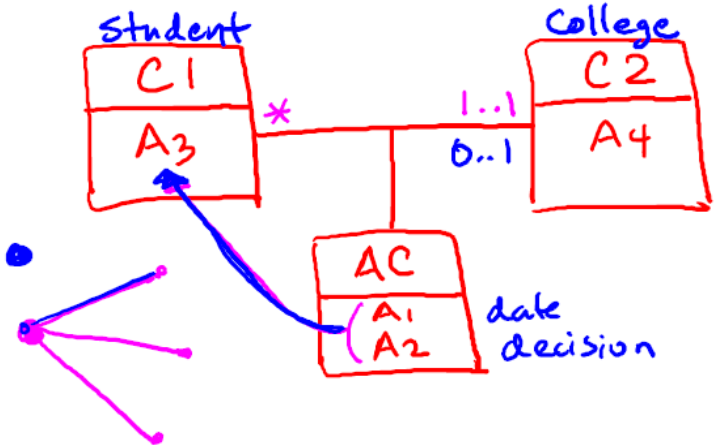
UML Data Modeling: **Association Classes**

Relationships between objects of two classes,
with attributes on relationships



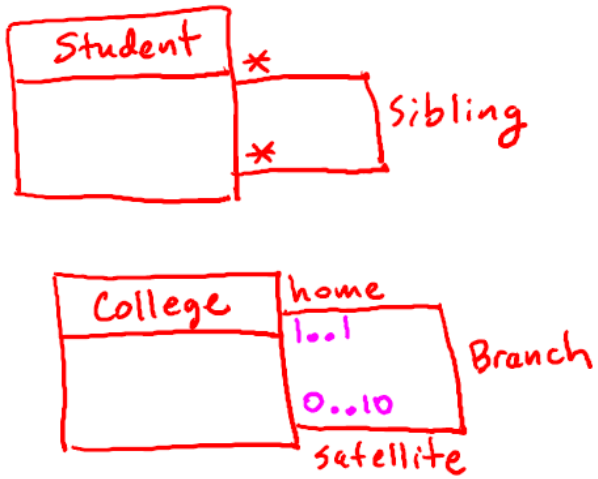
Eliminating Association Classes

Unnecessary if 0..1 or 1..1 multiplicity



Self-Associations

Associations between a class and itself



UML Data Modeling: 5 concepts

(1) Classes

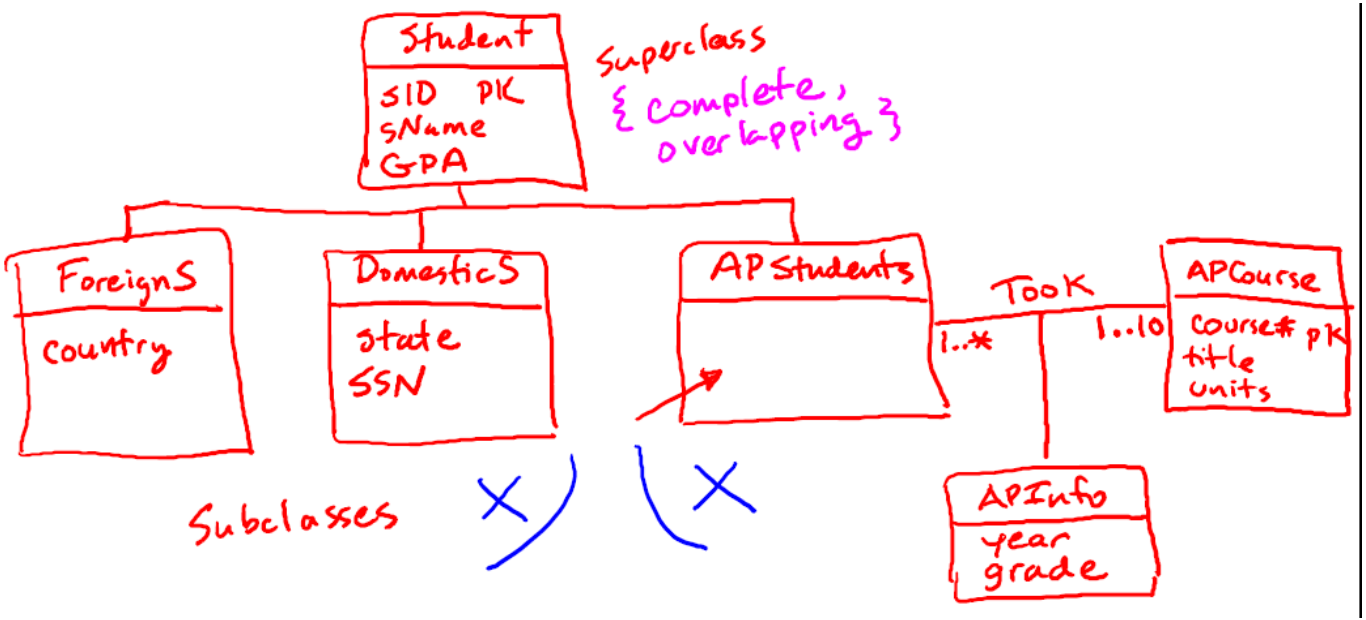
(2) Associations

(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

UML Data Modeling: **Subclasses**



Subclass Terminology & Properties

- *Superclass = Generalization*
- *Subclass = Specialization*

- *Incomplete* (*Partial*) vs. *Complete*
- *Disjoint* (*Exclusive*) vs. *Overlapping*

UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

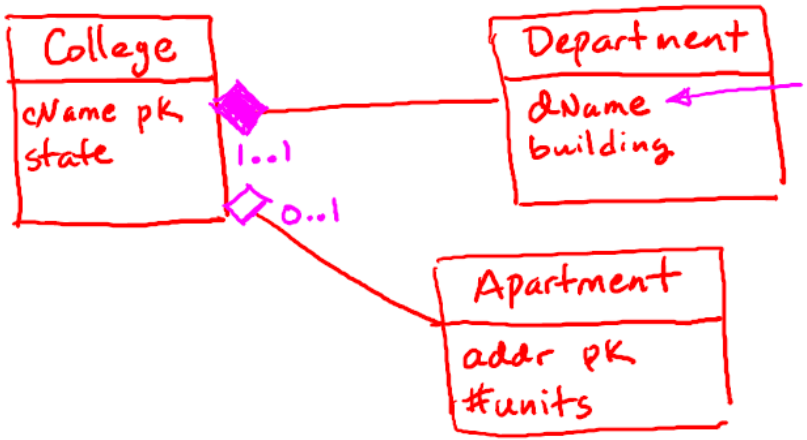
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

UML Data Modeling:
Composition & Aggregation

Objects of one class belong to objects of another class



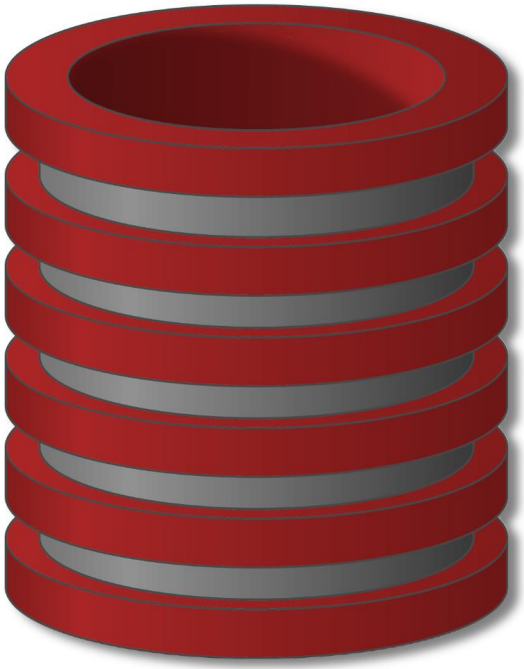
Higher-Level Database Design

- *Unified Modeling Language* (UML)

  Data modeling subset

- Graphical
- 5 concepts
  (1) Classes
  (2) Associations
  (3) Association Classes
  (4) Subclasses
  (5) Composition & Aggregation

❖ Can be translated to relations automatically

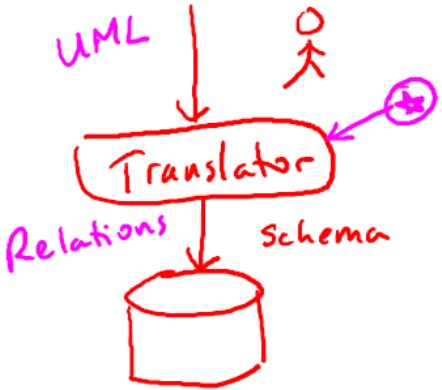


UML

UML to Relations

High-Level Database Design Model

- User-friendly (graphical) specification language
- Translated into model of DBMS



Unified Modeling Language (UML)

Data modeling subset

- 5 concepts
  (1) Classes
  (2) Associations
  (3) Association Classes
  (4) Subclasses
  (5) Composition & Aggregation
- Designs can be translated to relations automatically

  Provided every "regular" class has a key

UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

(3) Association Classes

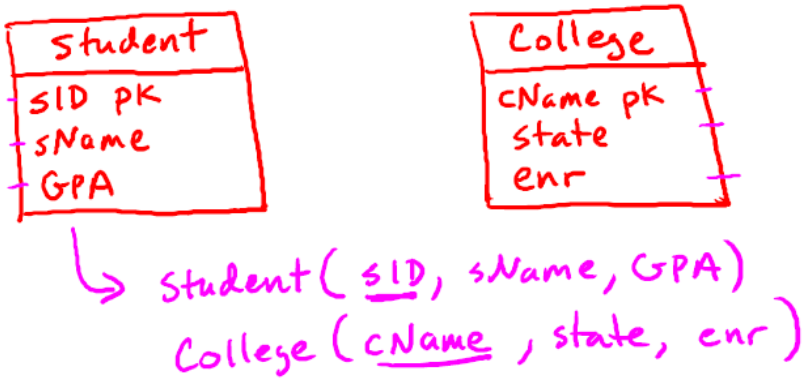
(4) Subclasses

(5) Composition & Aggregation

UML to Relations: **Classes**

Every class becomes a relation; pk → primary key

For data modeling: add "pk", drop methods



UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

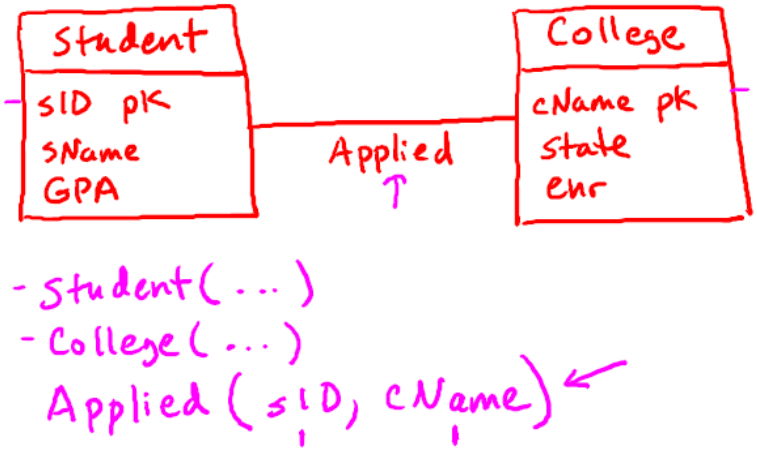
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

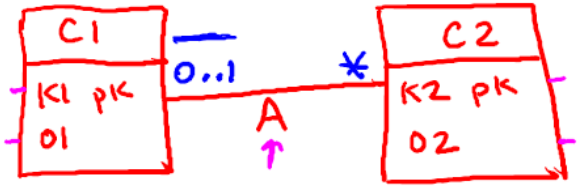
UML to Relations: **Associations**

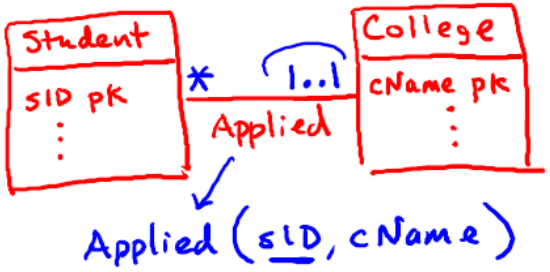
Relation with key from each side



Keys for Association Relations

Depends on multiplicity

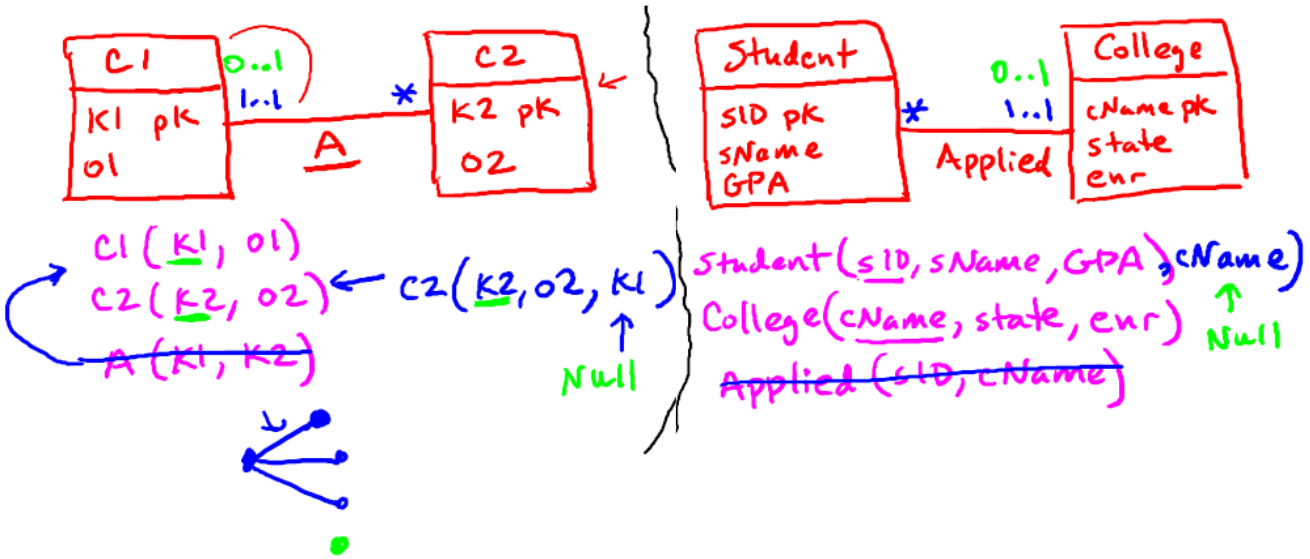




C1(K1, O1)
C2(K2, O2)
→ A(K1, K2) ←*

Association Relation Always Needed?

Depends on multiplicity



UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

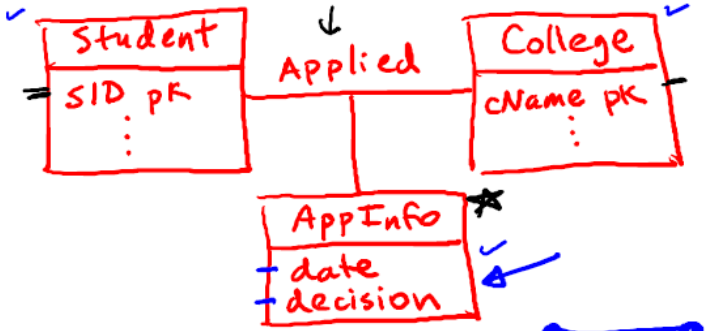
(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

UML to Relations: **Association Classes**

Add attributes to relation for association

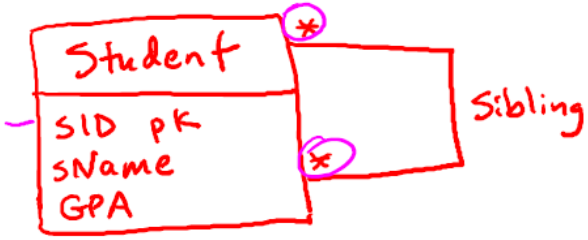


Require a key for every "regular" class.

Determining keys
"Folding"

Student(sID, ...)
College(cName, ...)
Applied(sID, cName, date, decision)

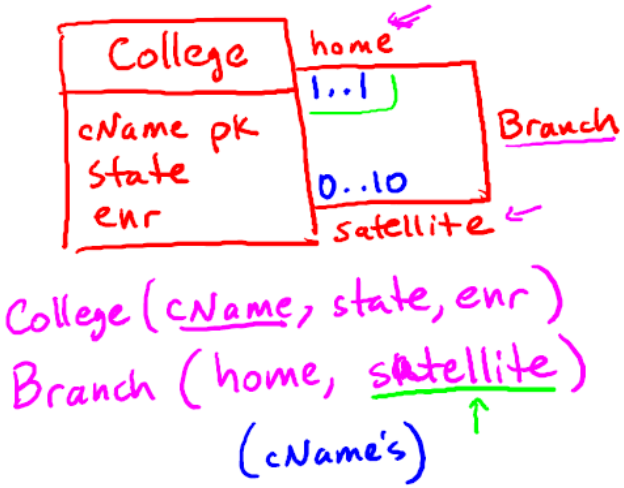
Self-Associations



Student(sID, sName, GPA)

Sibling(sID1, sID2) ←

Self-Associations



UML Data Modeling: 5 concepts

(1) Classes

(2) Associations

(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

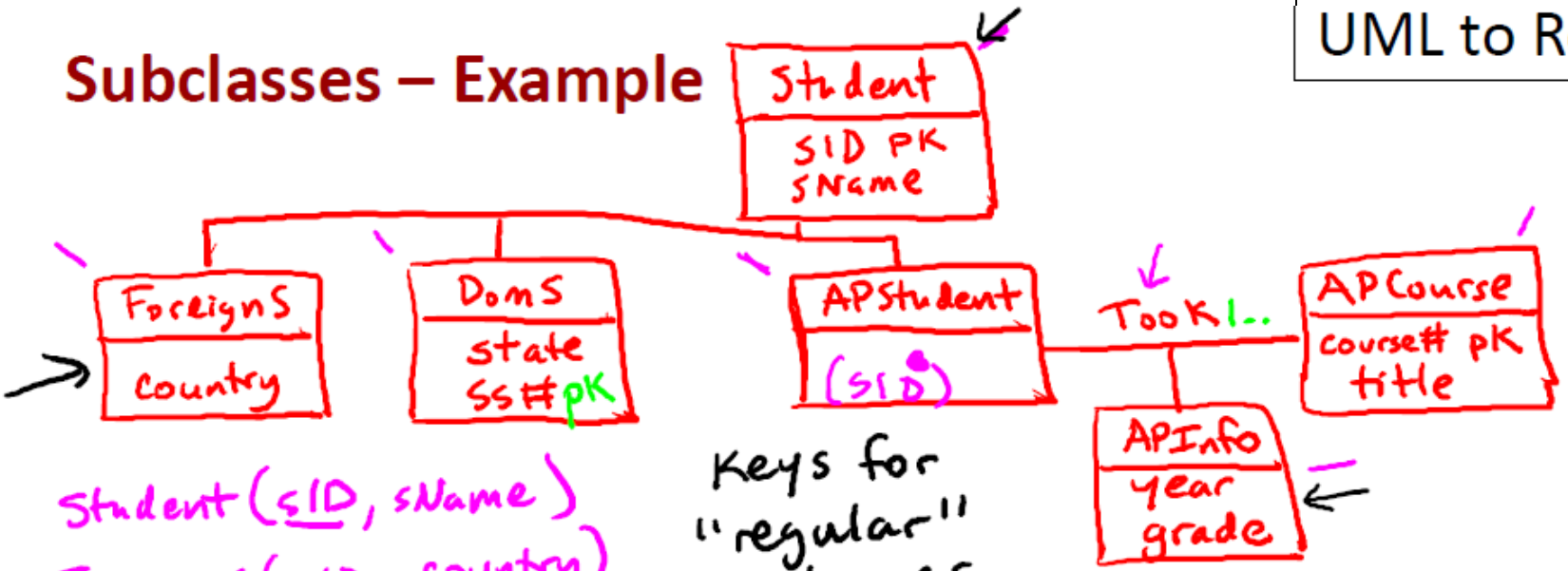
Subclasses

1) Subclass relations contain superclass key + specialized attrs.
2) Subclass relations contain all attributes
3) One relation containing all superclass + subclass attrs.

Best translation may depend on properties

(1) S(K, A) S1(K, B) S2(K, C)
(2) ~~S(K, A)~~ S1(K, A, B) S2(K, A, C)
(3) S(K, A, B, C)

Subclasses – Example



Keys for "regular" classes

Student(<u>SID</u>, SName)

ForeignS(<u>SID</u>, country)

DomS(<u>SID</u>, state, <u>SS#</u>)

APStudent(<u>SID</u>)

APCourse(<u>course#</u>, title)

TookI(SID, course#, year, grade)

UML Data Modeling: 5 concepts

(1) Classes

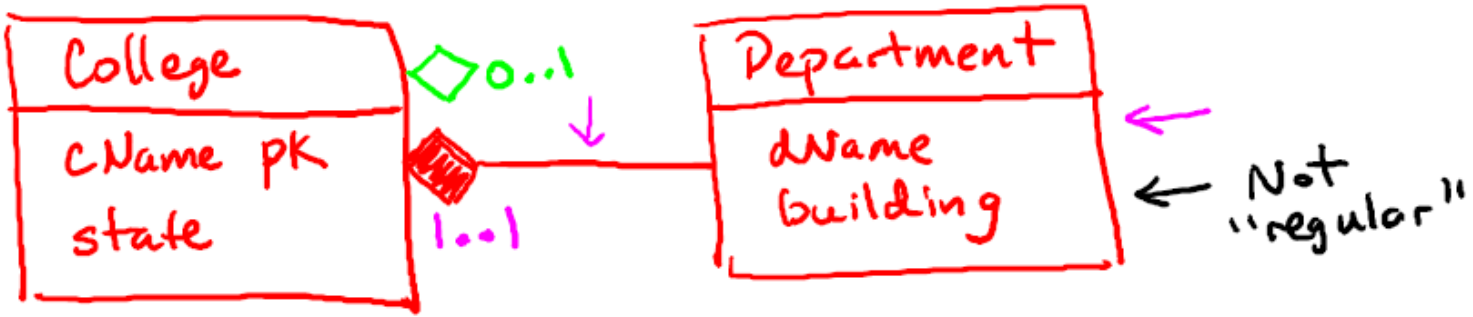
(2) Associations

(3) Association Classes

(4) Subclasses

(5) Composition & Aggregation

Composition & Aggregation



College(cName, state)

Department(dName, building, cName)

Null

UML: High-Level Database Design Model

- User-friendly graphical specification language
- Designs translated to relations automatically