

Introduction to Database Concept

DATA vs INFORMATION

- Data
 - Raw facts
 - Not processed
 - Ex. : score from exam.
- Information
 - Processed raw data
 - Ex. : the average score

Why data is important?

- Data is building block for information.
 - And it keeps getting larger and larger
- Accurate, relevant and timely information is the key to good decision.
- Good decision leads to benefits.
- Data management is important.

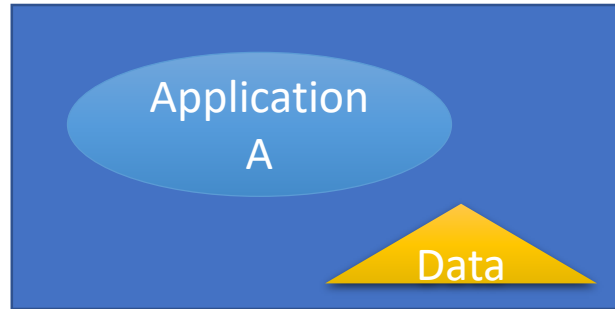


Traditional File-Based System

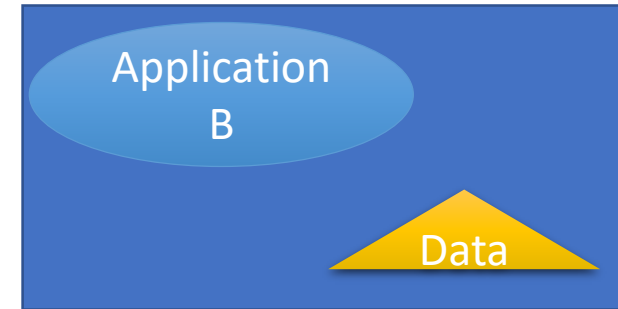
“A collection of application programs that perform services for the end-users such as the production of reports. **Each program defines and manages its own data.**”

Connolly, T. M., & Begg, C. E. (2005). *Database systems: a practical approach to design, implementation, and management*. Pearson Education.

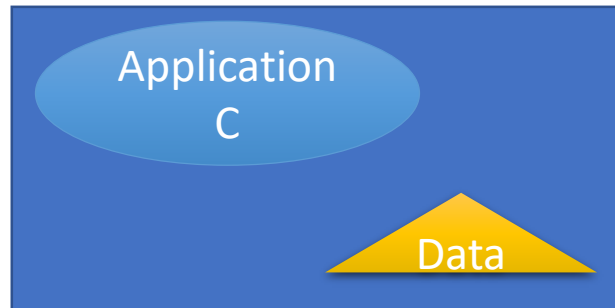
Traditional File-Based System



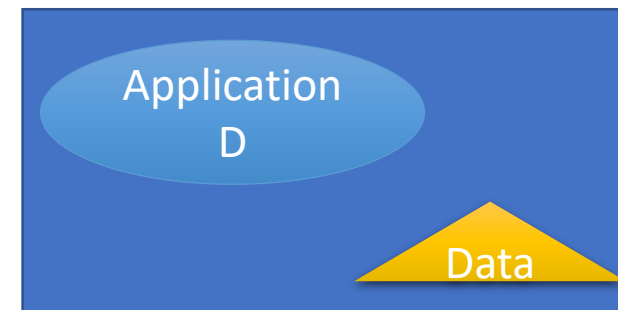
Department A



Department B



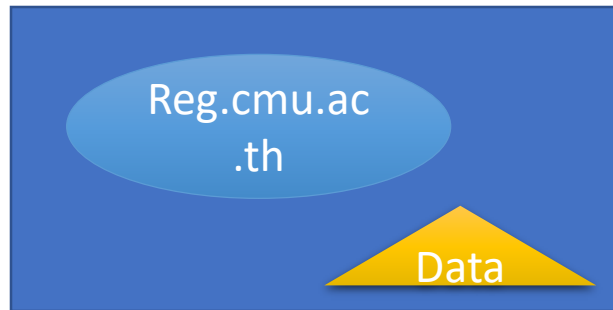
Department C



Department D

Limitation of the file-based system

- Separation and isolation of data



Registration



SE Department

How can an advisor get the list of the enrolled courses for an a

Limitation of the file-based system

- Duplication of data



Registration

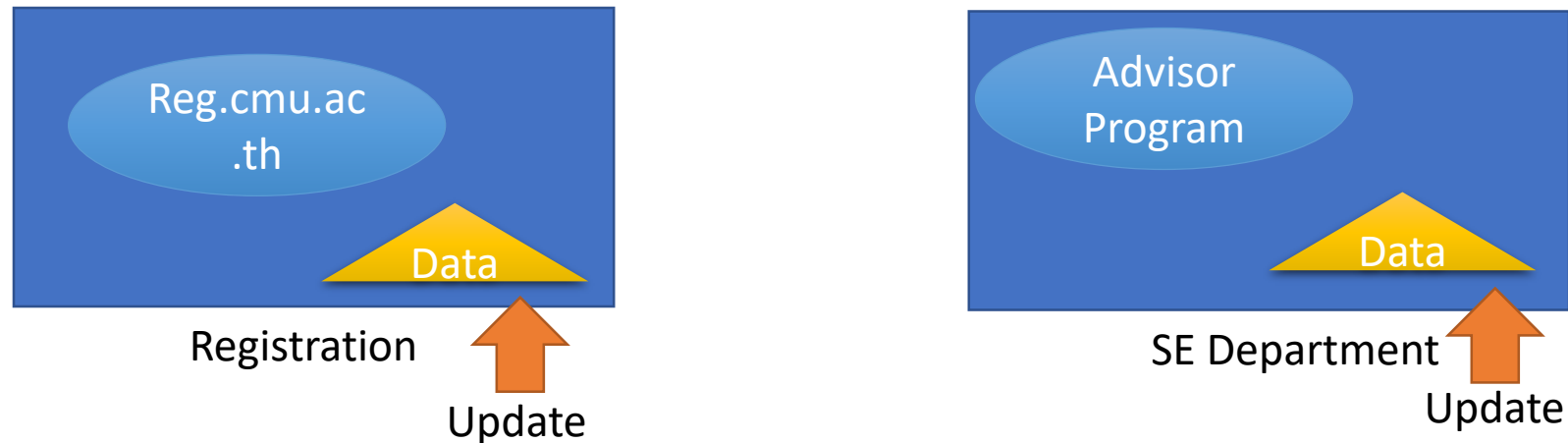


SE Department

Why the student data has to be in 2 places?

Limitation of the file-based system

- Data dependence



What if the student changes the name?

Limitation of the file-based system

- Incompatible file formats



Limitation of the file-based system

- Fixed query.
 - Each application depends on the developer.
 - There is no room for unplanned query.

QUERY = INQUIRY =
QUESTIONING

Limitation of the file-based system

- Separation and isolation of data
- Duplication of data
- Data dependence
- Incompatible file formats
- Fixed query

Database approach

- The source of the problem of the filed-based system are :
 - 1) The data is embedded in the application.
 - 2) No mechanism other than the developed application to control the data.
- The databased approach is devised to overcome the problem.

Database approach

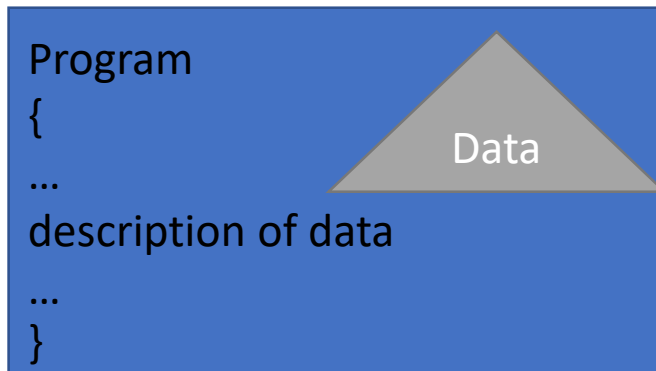
- Definition of database :

“A **shared collection of logically related data** and a description of this data designed to meet the information needs of an **organization.**”

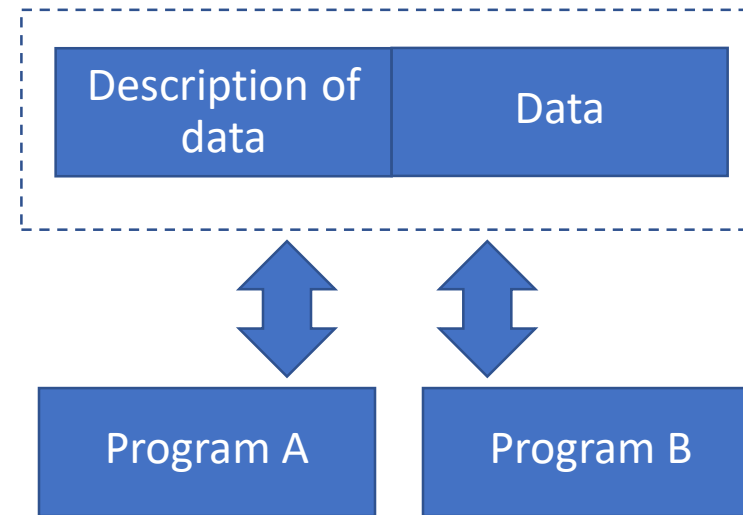
Cornduff, T. M., & Beegh, C. E. (2005). *Database systems: a practical approach to design, implementation, and management*. Pearson Education.

Characteristics of database approach

- Self-describing nature of a database system
 - Contain both data and description of data
 - In the file-based system, the description of data is part of the application.
 - No one can use it, except the program.



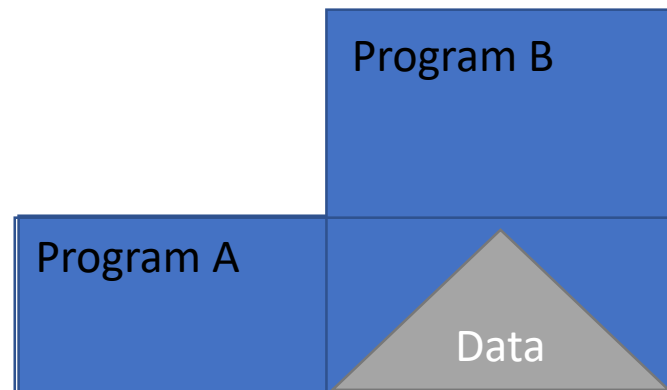
file-based system



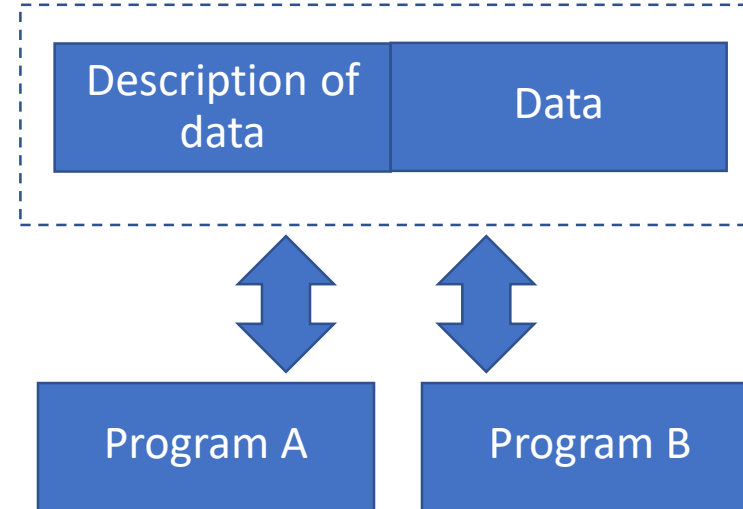
database system

Characteristics of database approach

- Insulation between programs and data, data abstraction
 - The program and the program is separated.
 - In the file-based system, the description of data is part of the application.
 - If you make a change in description of data, you have to change all the program.



file-based system

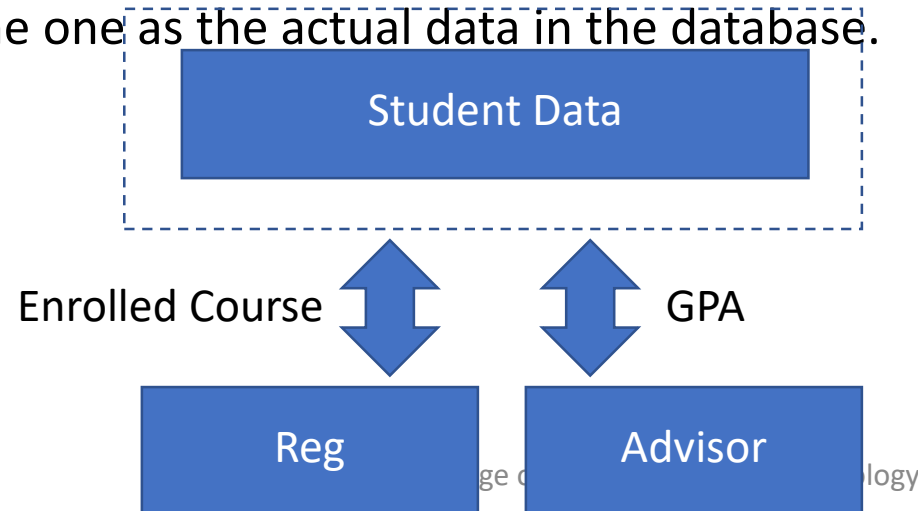


database system

Characteristics of database approach

- Support multiple views of the data
 - There can be many user.
 - Each user may need a different set of information.
- A view is a virtual data.

- Not same one as the actual data in the database.



Characteristics of database approach

- Sharing of data and supporting multiple user
 - Provide concurrency control
 - Allows many user to use at the same time.
 - Think about the cinema ticket or flight ticket.
- Transaction
 - An execution of program with the database.
 - Reading, updating the data in the database
 - Each transaction is isolate from each other transaction.

Characteristics of database approach

- Self-describing nature of a database system
- Insulation between programs and data, data abstraction
- Support multiple views of the data
- Sharing of data and supporting multiple user

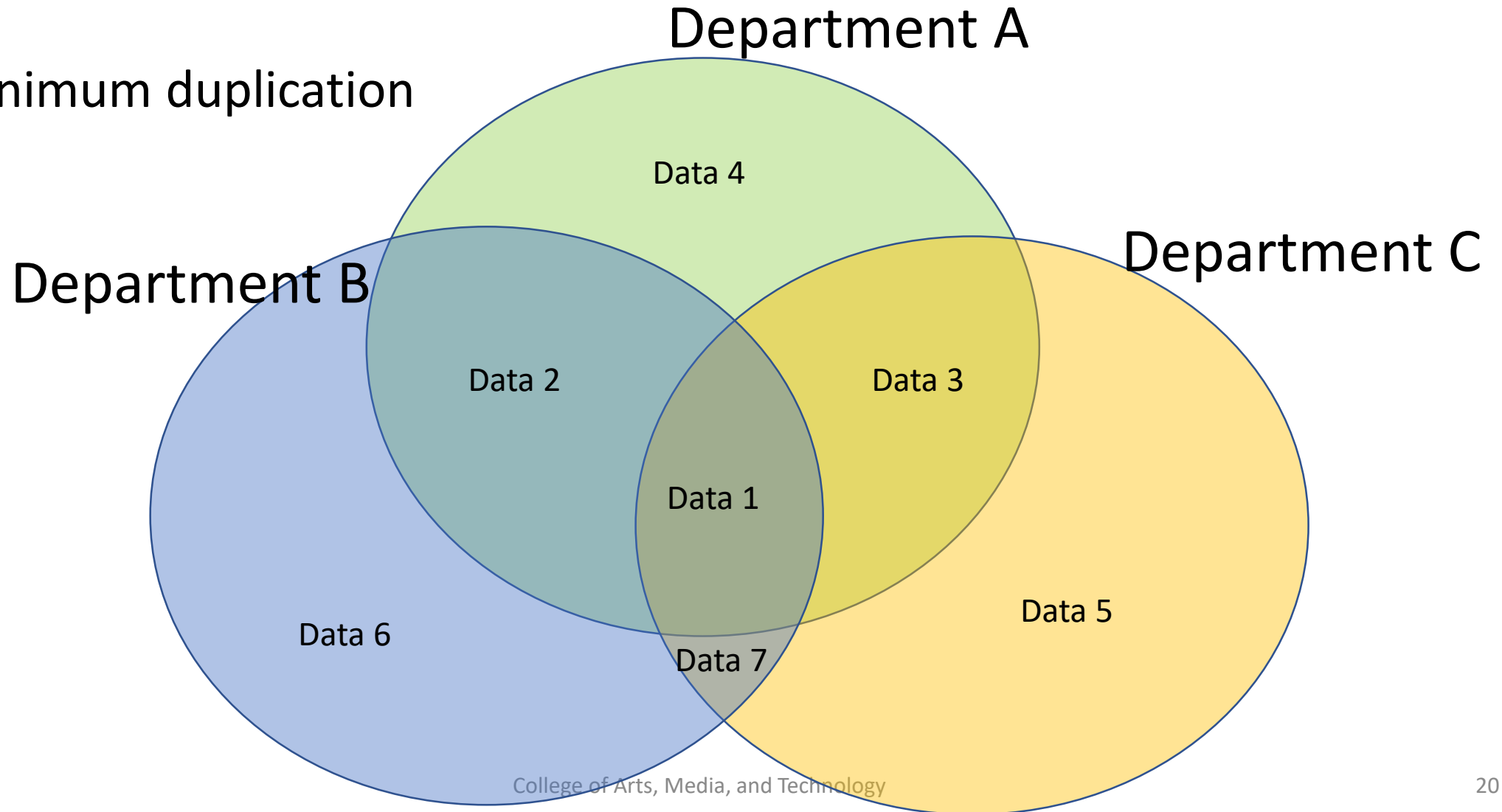
Benefits of Database approach

- Single large repository of data



Benefits of Database approach

- Minimum duplication



Benefits of Database approach

- Minimum duplication
 - When each department uses the same data, it does not have to be different data.
 - It can use the same one.
 - Ex. Data 2 is used by Department A and Department B.

Department A = Data 1 + Data 2 + Data 3 + Data 4

Department B = Data 1 + Data 2 + Data 6 + Data 7

Department C = Data 1 + Data 3 + Data 5 + Data 7

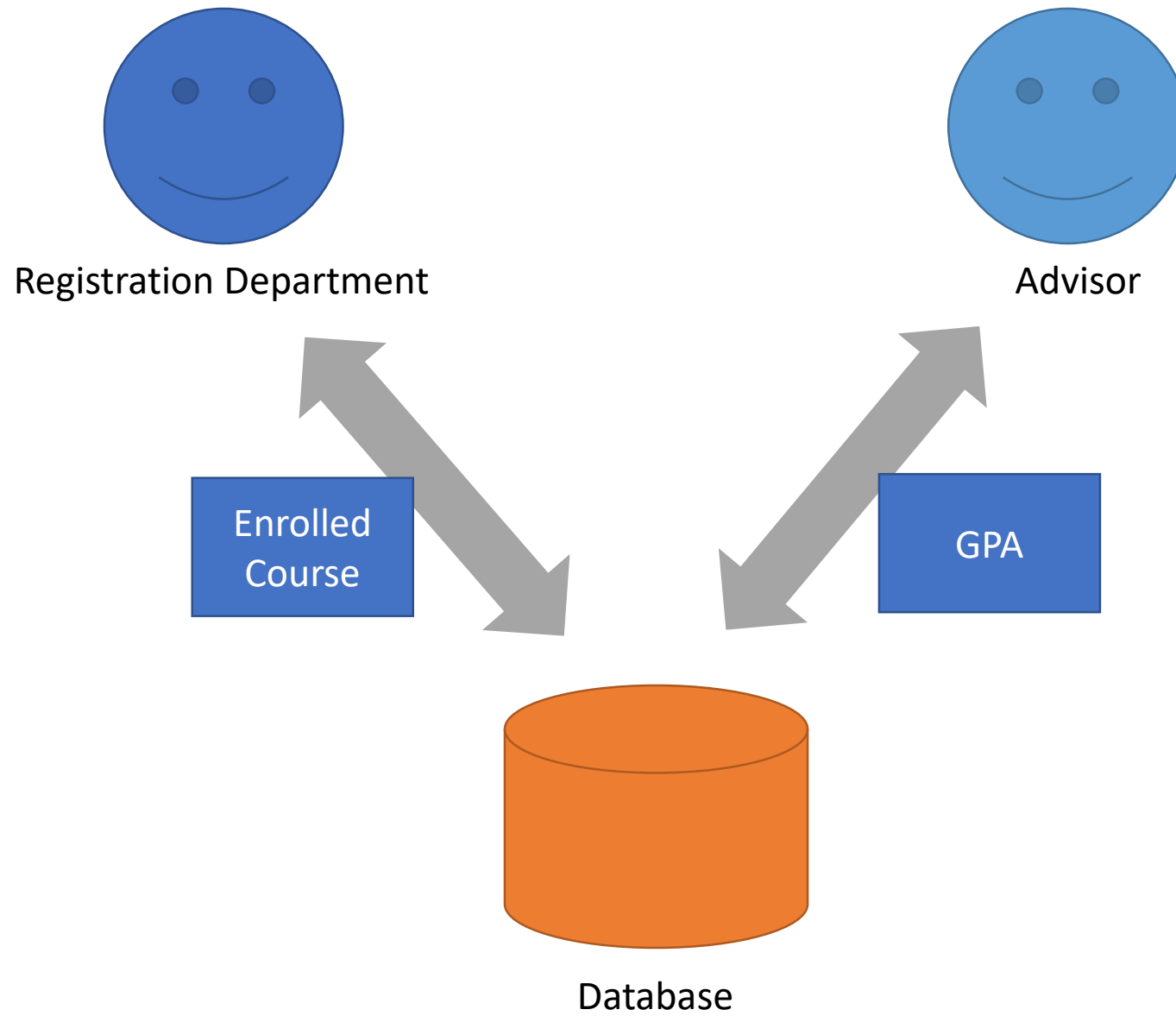
Benefits of Database approach

- Single large repository of data
- Minimum duplication of data
- Etc.

Database concept

Introduction

- User from different department needs different data.
- By the use of the database, the data is stored in one place with the manipulation mechanism.
- The database system provides the **abstract view** of data to the user.
 - Hide the detail of how data is stored and manipulated.
 - Show each user what they want



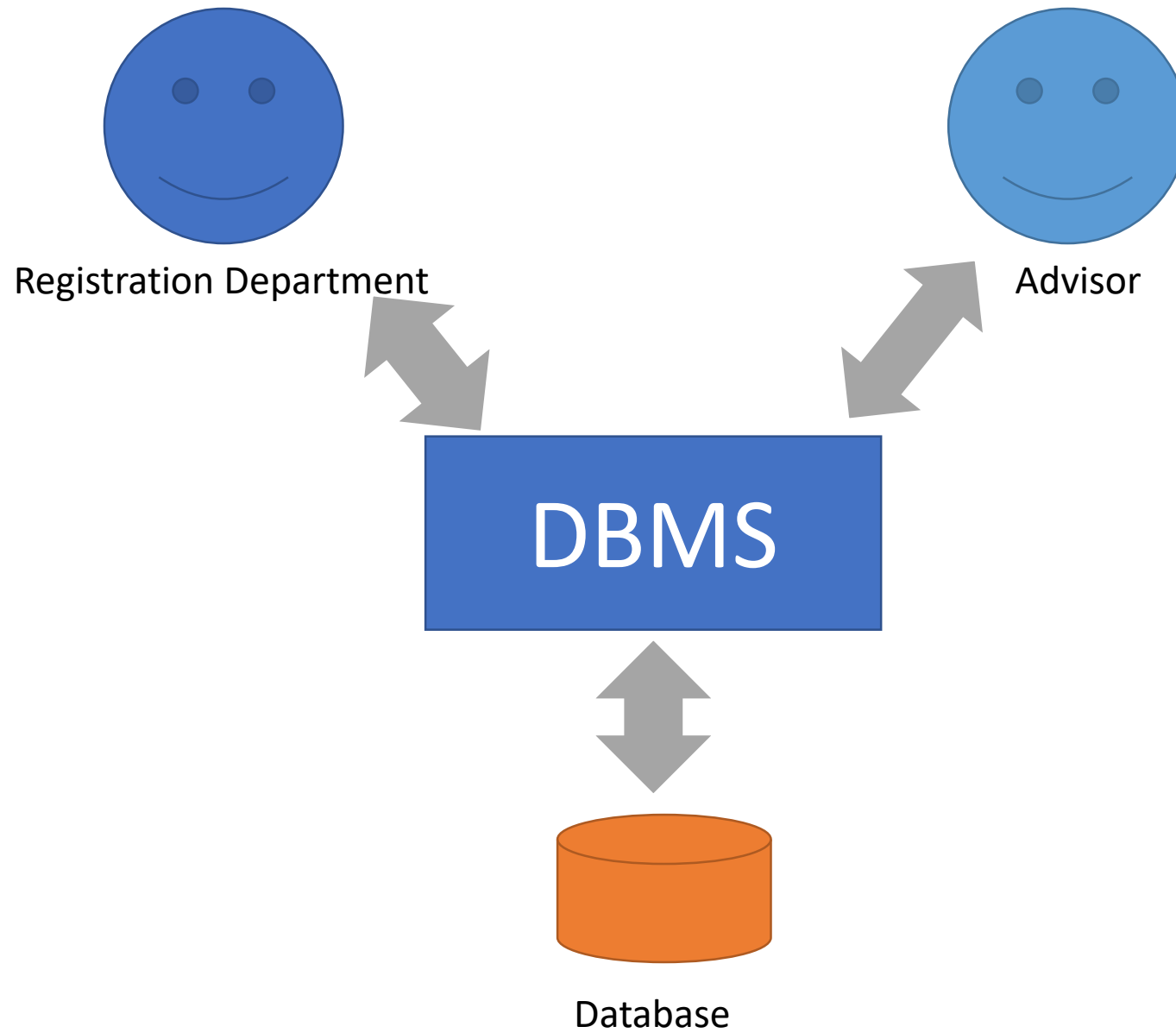
In this chapter

- The definition and component of the DBMS
- The architecture of database system

Database Management System

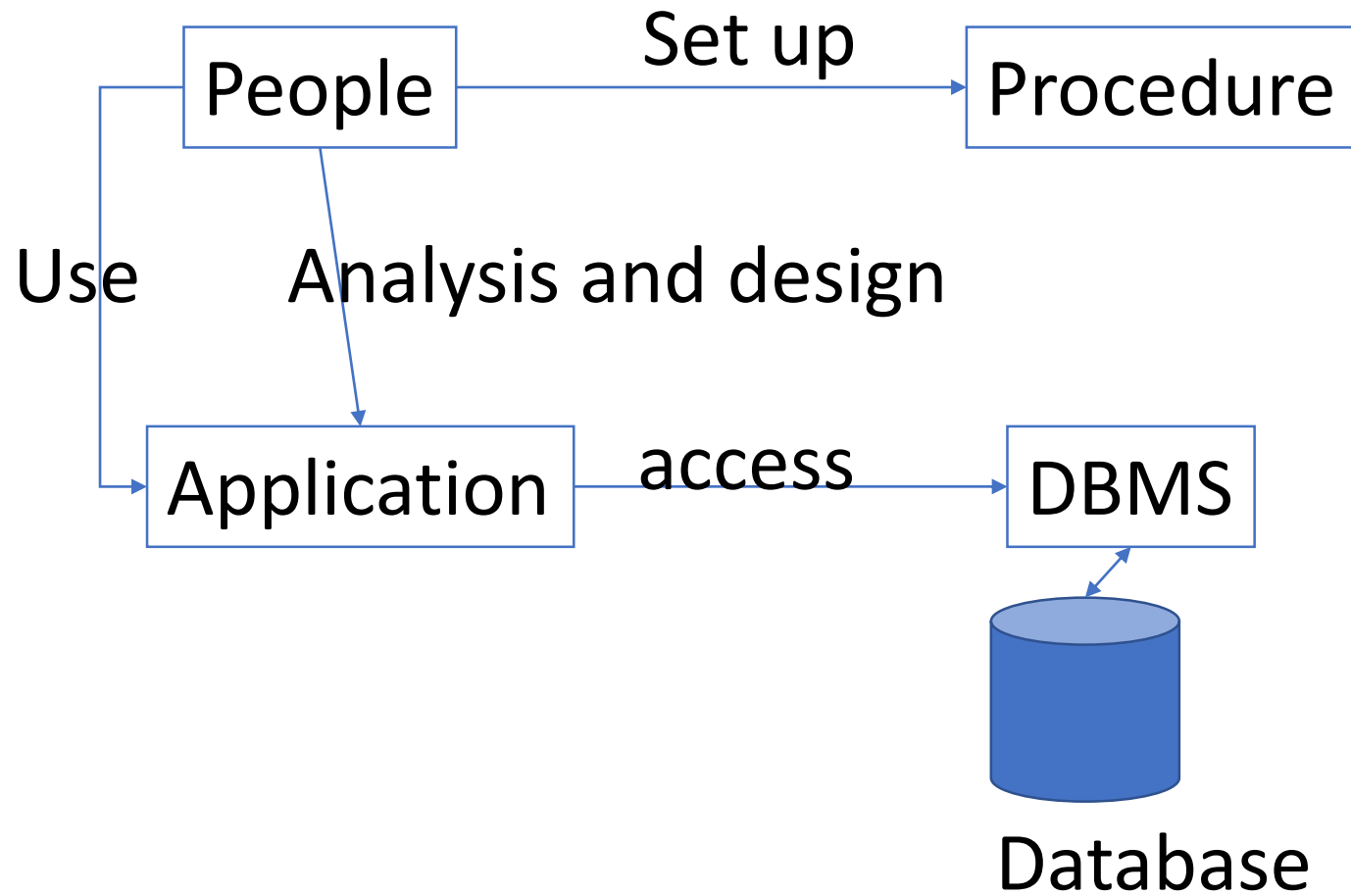
- Database Management System (DBMS)

“A software system that enables users to define, create, maintain and controlling access to the database.”

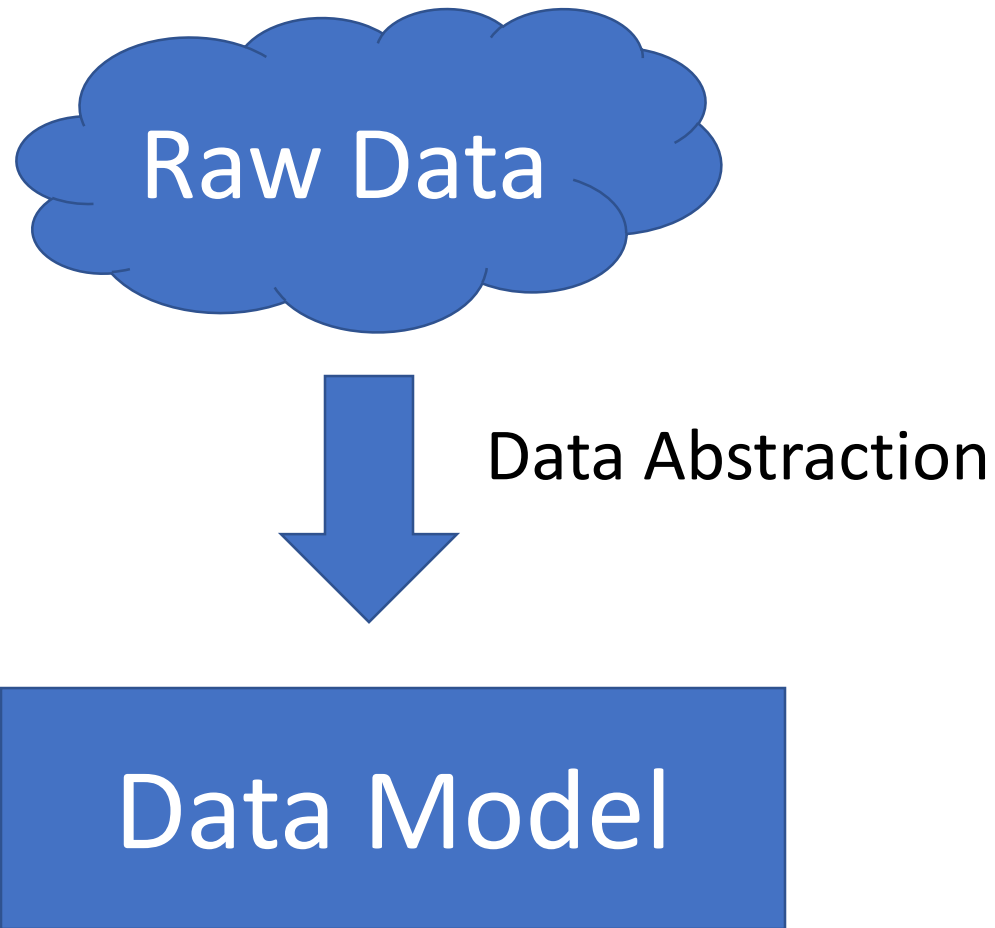


Environment of DBMS

- Hardware
 - Needs hardware to run
 - Single computer to network of computer
- Software
 - DBMS, itself
 - OS, networking sw, ...
- Data
- Procedure
 - Rule to govern the design and usage
- People
 - Database administrator
 - Database designer
 - System analyst
 - End user



Data models, Schemas, and Instance

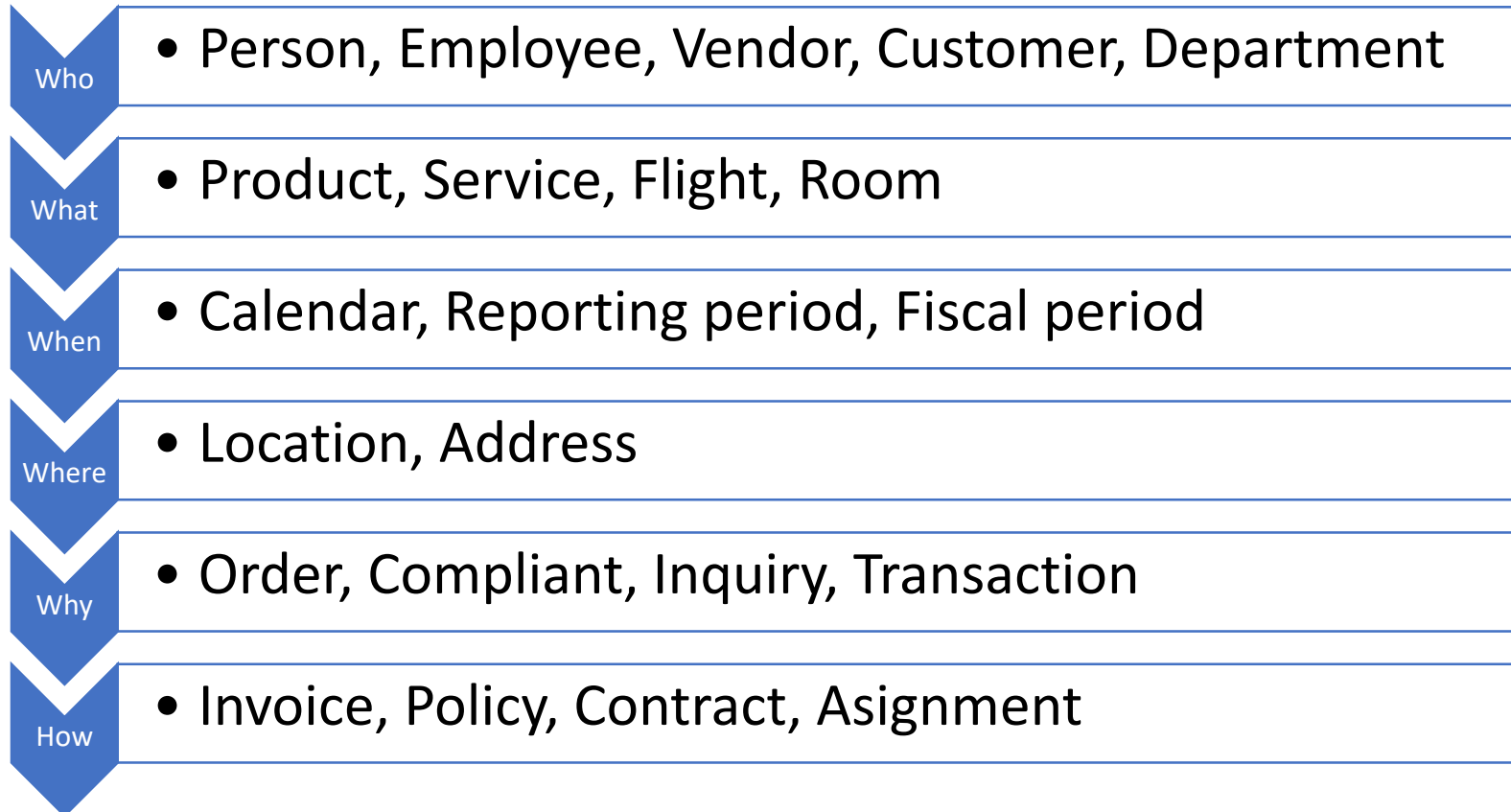


Data models, Schemas, and Instance

- Data Abstraction
 - *“The suppression of the details of data organization and storage and highlighting of the essential features for an improved understanding of data.”*
- Data Model
 - *“A collection of concepts that can be used to describe the structure of database.”*

Data models, Schemas, and Instance

Data model represents



Real World



Student

Digital World

Student

Student ID
First name
Last name
Tel.
Citizen ID
Address

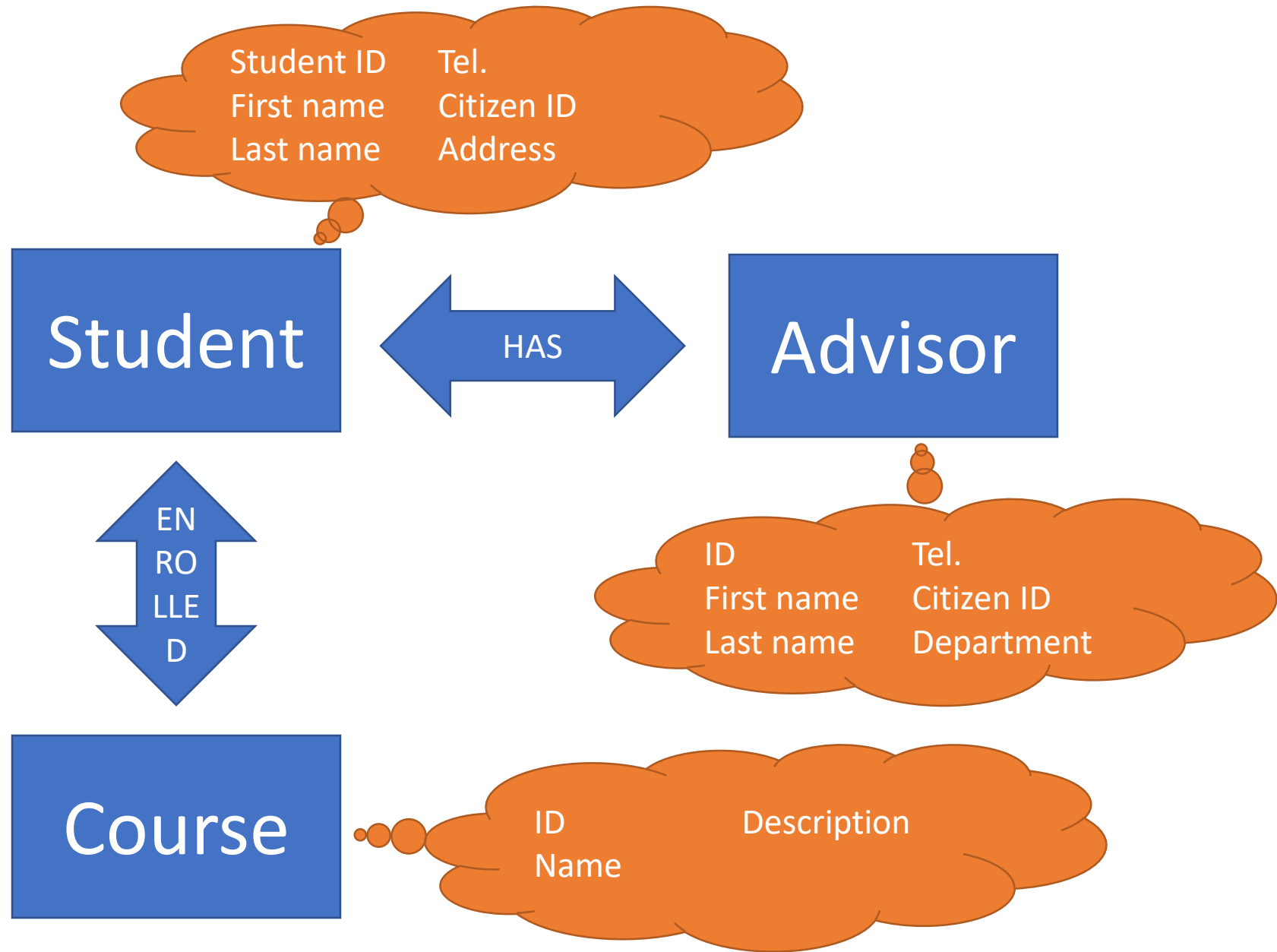
Advisor

ID
First name
Last name
Tel.
Citizen ID
Address
Department

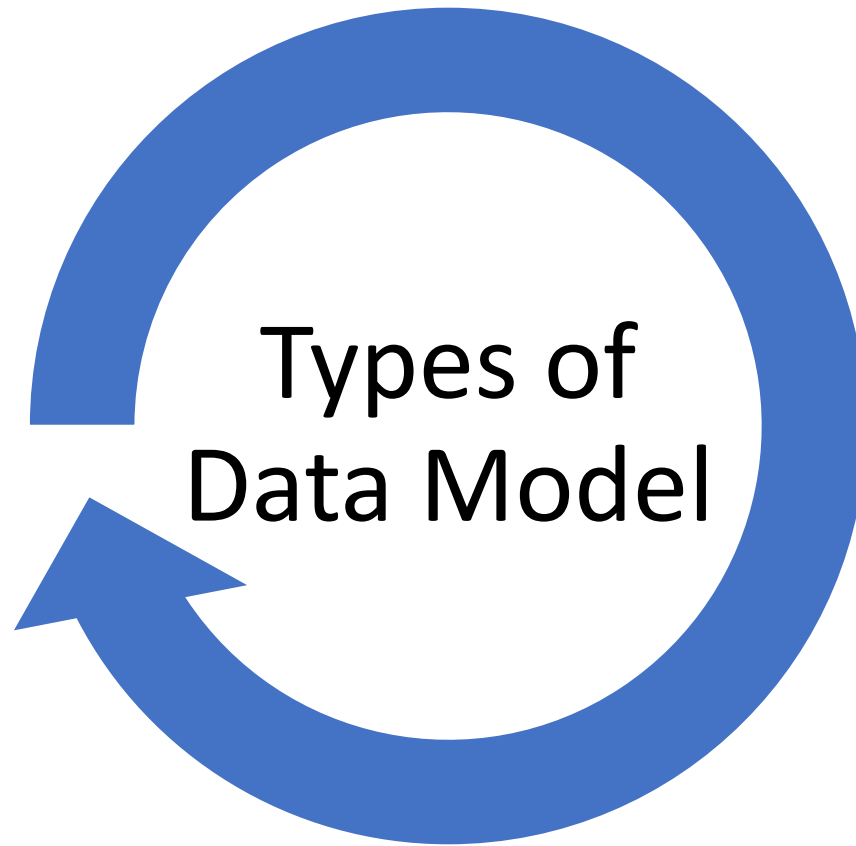
Course

ID
Name
Description
Department

Data Abstraction



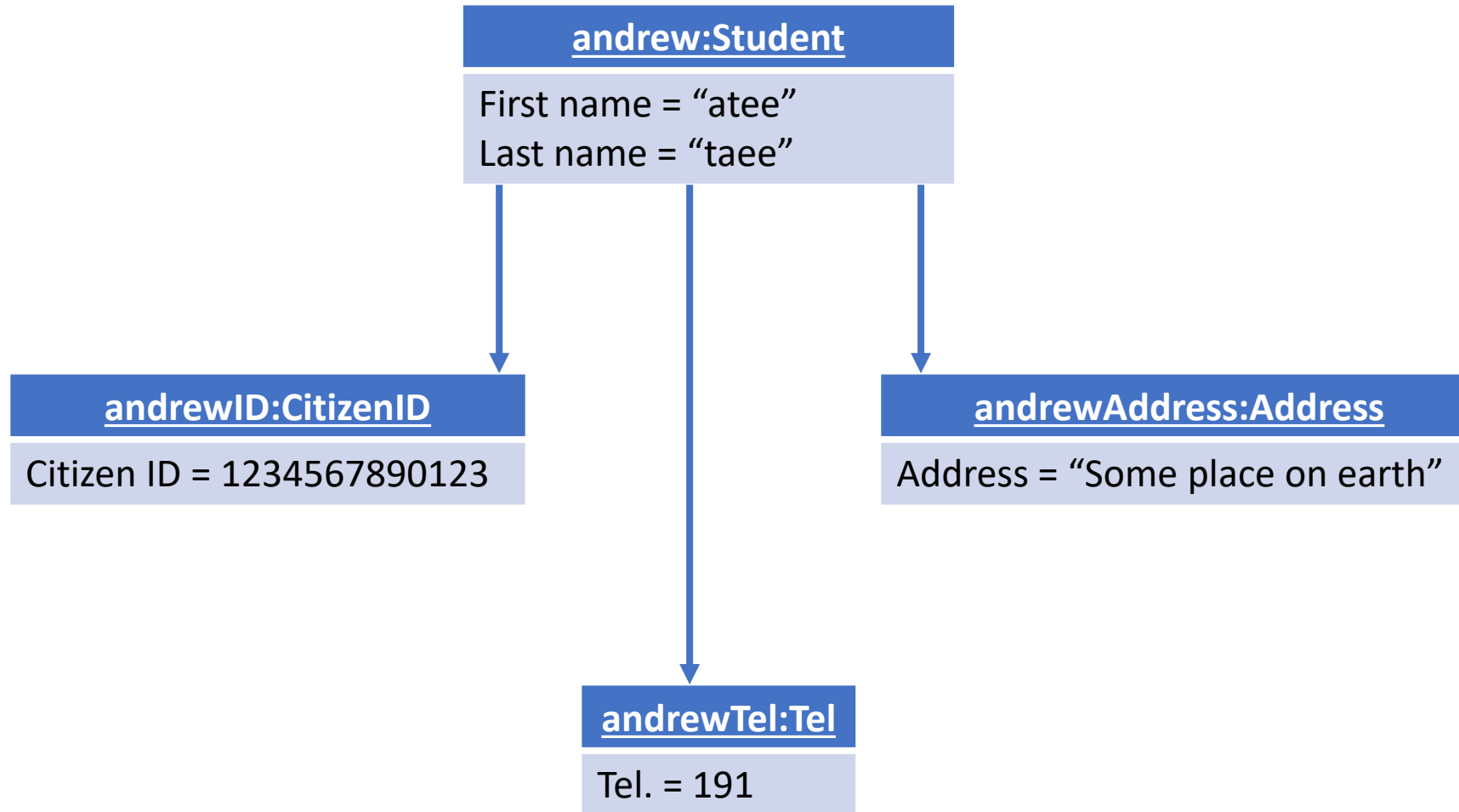
Categories of Data Models



- Object-based
- Record-based
- Physical

Categories of Data Models

- Object-Based data models
 - Model the data using entities, attributes and relationships.
 - The entity is a object in the context
 - Person, place, things, concepts, event
 - The attribute is a property that describes some aspect of the entity.
 - Color of flower, student ID of a student, address of a teacher
 - The relationship is an association between entities.
 - A student live at an address.
 - Room locates in an building.



Categories of Data Models

- Record-Based data models
 - The data is modeled by a number of fixed format records.
- There are 3 types of Record-Based data models
- Relational data model
 - Represent by table where column is a unique name.
 - We will dig deeper in the next chapter

Categories of Data Models

sID	First name	Last name
1	atee	taee
2	trrrr	rtttt

sID	Tel	sID
1	191	1
2	9911	2

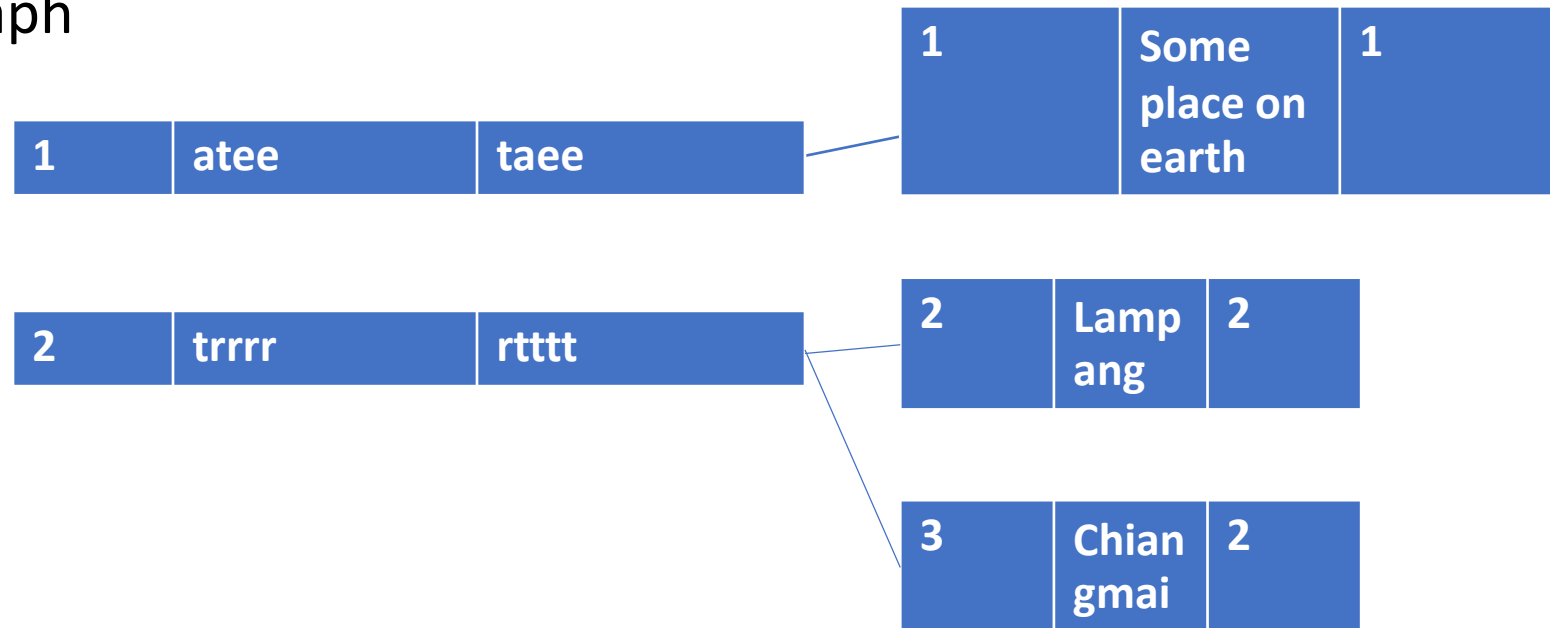
cID	Citizen ID	sID
1	12345678 90123	1
2	22222222 2	2

aID	Address	sID
1	Some place on earth	1
2	Lampang	2
3	Chiangmai	2

- Relational data model

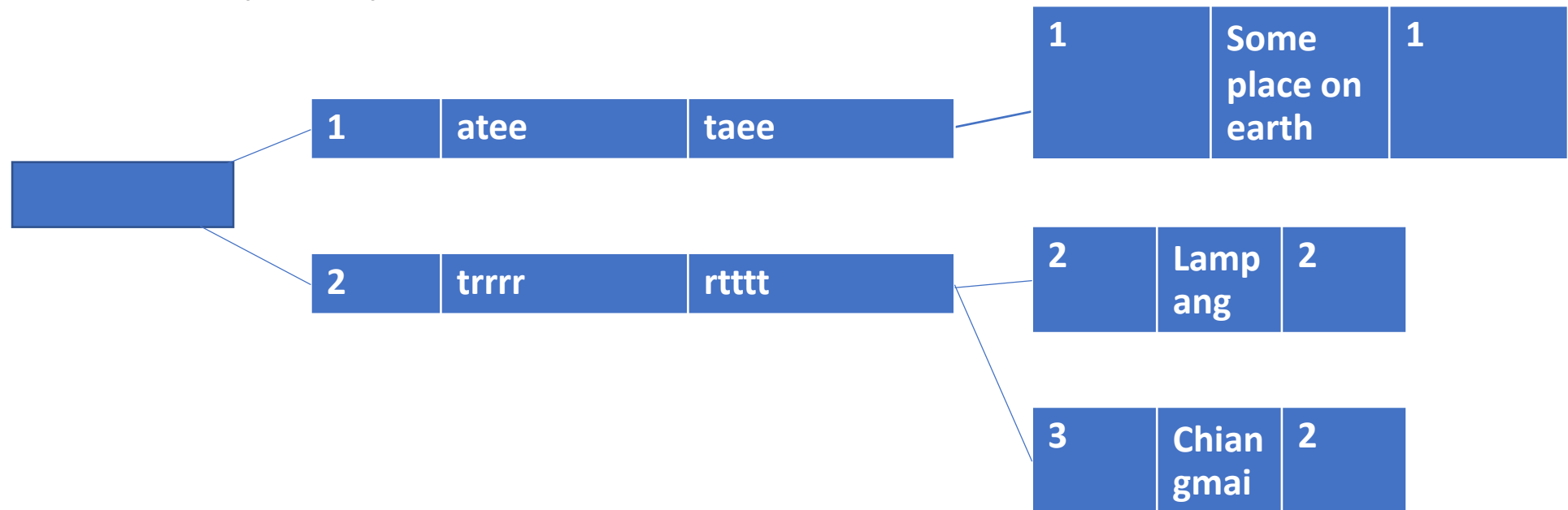
Categories of Data Models

- Network data model
 - The record is linked together by edges
 - Similar to graph



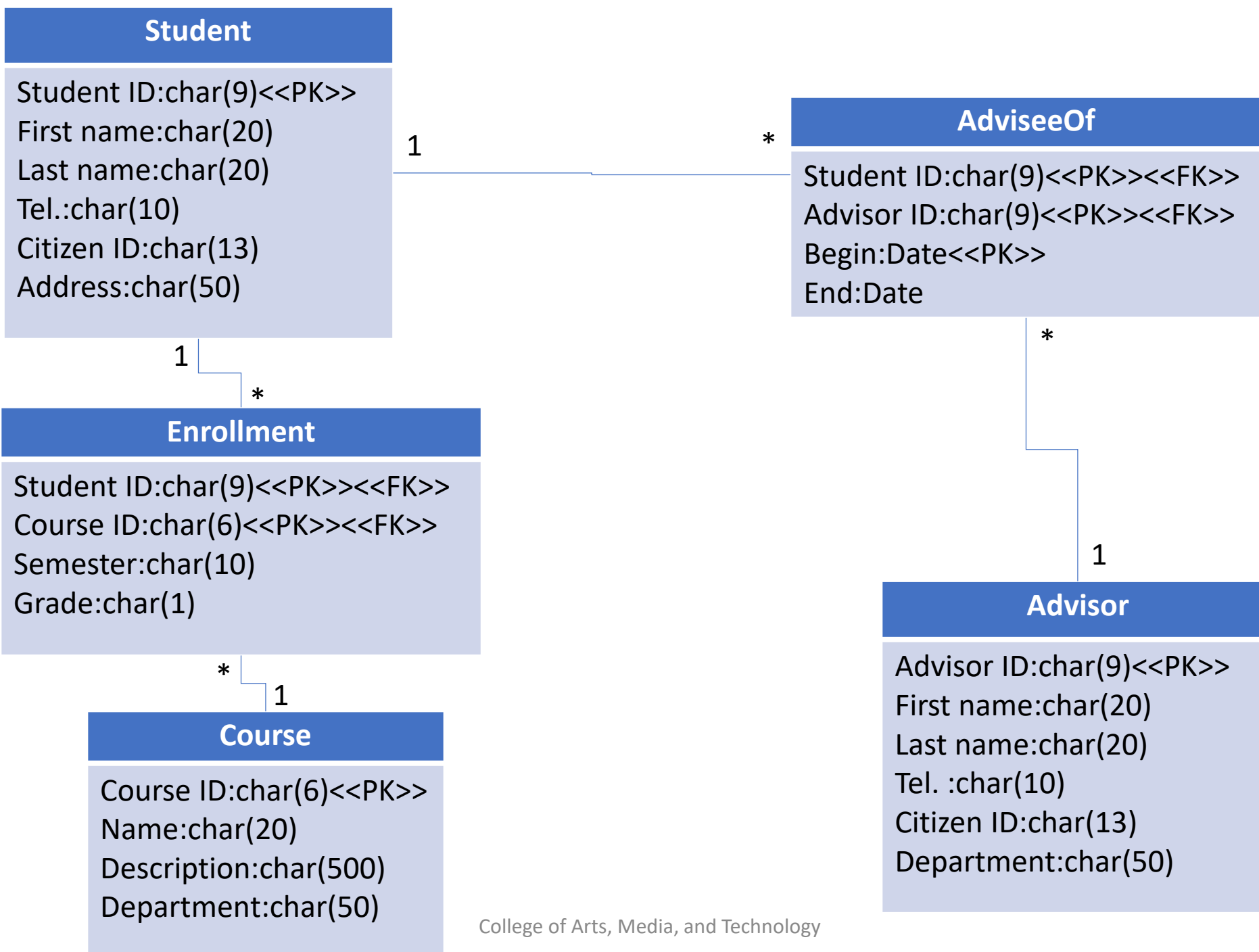
Categories of Data Models

- Hierarchical data model
 - Similar to the network data model
 - Allows only one parents

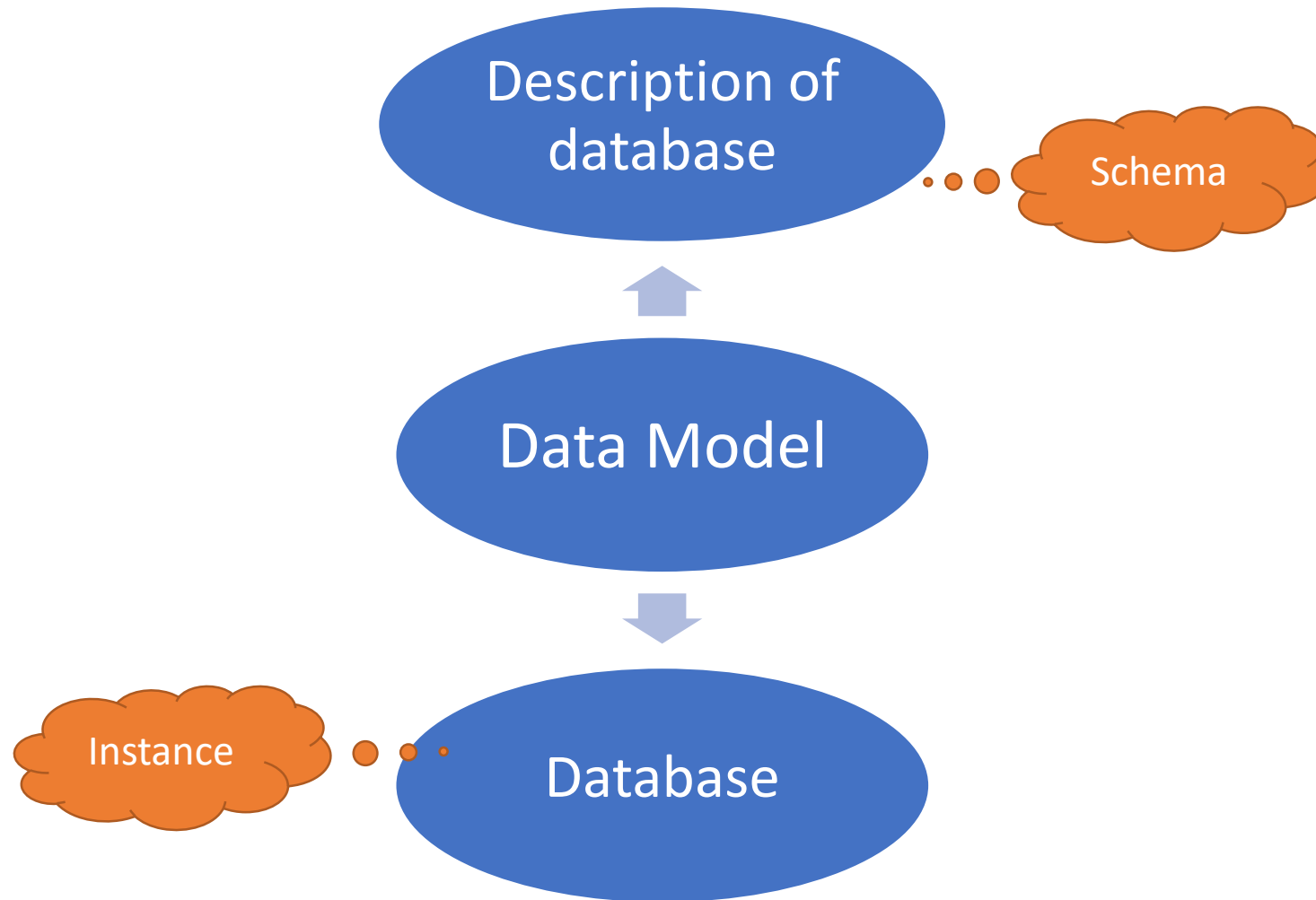


Categories of Data Models

- Physical data models
 - Sometimes, it is referred as low-level data models
 - Describe how the information store in the computer
 - Not for business user, for computer specialist



Schemas, Instance, and Database State




Schemas, Instance, and Database State

- Database schemas
 - The description of database.
 - Design process
 - Not change frequently
- Database instance
 - Sometimes, it is referred as database state or database snapshot
 - Actual data in the database at a particular time
 - Change frequently

Schemas, Instance, and Database State

Schemas




Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

{StudentID, First name, Last name, Tel, Citizen ID, Address}

Schemas, Instance, and Database State

Instance



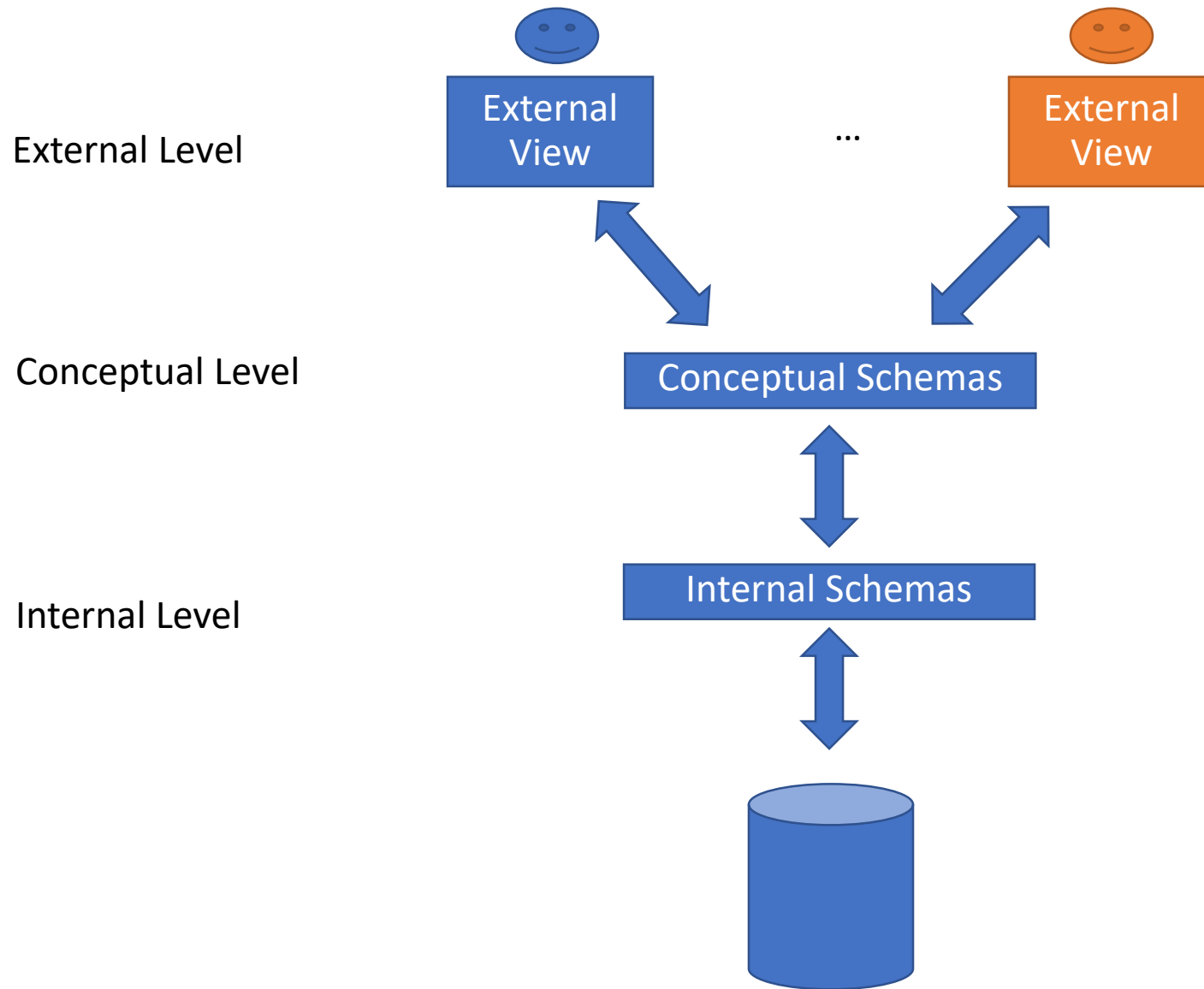
Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

{590551020,Somchai,Somwang,1199,111111111,Lamphun}

{590551021,Somying,Maisomwang,9911,222222222,Lampang}

Three-Schema Architecture

- Recall the characteristics of the database
 - Self-describing nature of a database system
 - Insulation between programs and data, data abstraction
 - Support multiple views of the data
- Combine the characteristics of the database together.



Three-Schema Architecture

- External Level

“The users’ view of the database. This level describes that part of the database that is relevant to each user.”

- Includes only the entities, attributes and relationships that a user needs.

Three-Schema Architecture

- Conceptual Level
 - “The community view of the database. This level describe **what** data is stored in the database and the relationships among the data.”
- Logical structure of the **entire** database.
- Not storage dependent.

Three-Schema Architecture

- Internal Level

“The physical representation of the database on the computer. This level describes **how** the data is stored in the database.”

- Include data structure and file organization.

- Not physical level

External Level

StudentID	First name	Last name	Tel	Address
-----------	------------	-----------	-----	---------

StudentID	AdvisorID	Advisor name
-----------	-----------	--------------

Conceptual Level

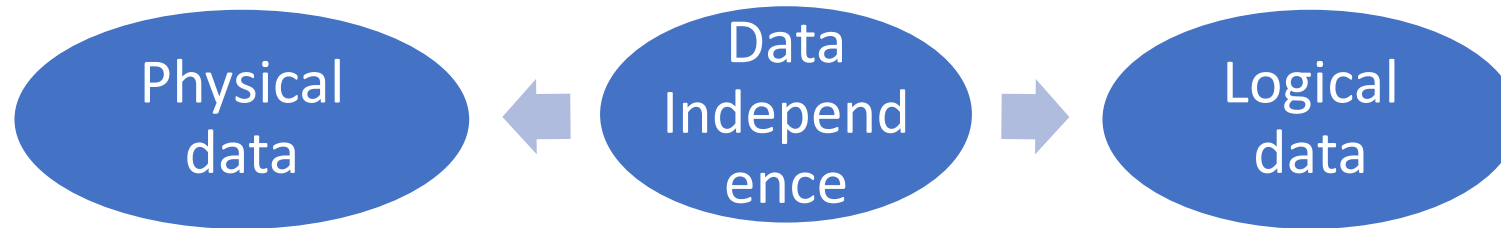
StudentID	First name	Last name	Tel	Address	AdvisorID	Advisor name
-----------	------------	-----------	-----	---------	-----------	--------------

Internal Level

```
public class Node{  
    String StudentID;  
    String Firstname;  
    String Lastname;  
    String Tel;  
    String Address;  
    String AdvisorID;  
    String AdvisorName;  
    Node next = new Node();  
    ...  
}
```

Data Independence

The three-schema architecture supports 2 types of data independence



Data Independence

- Logical data independence
 - The change in conceptual level does not effect the external level.
- The new data can be added to the conceptual data with out effecting the external level.
 - The data in the external level still use the information from the original schema.
- Except the external data want the newly added schemas.

External Level

StudentID	First name	Last name	Tel	Address
-----------	------------	-----------	-----	---------

StudentID	Advisor ID	Advisor name
-----------	------------	--------------

Conceptual Level

StudentID	First name	Last name	Tel	Address	AdvisorID	Advisor name	Student Age
-----------	------------	-----------	-----	---------	-----------	--------------	-------------

Internal Level

```
public class Node{  
    String StudentID;  
    String Firstname;  
    String Lastname;  
    String Tel;  
    String Address;  
    String AdvisorID;  
    String AdvisorName;  
    Node next = new Node();  
    ...  
}
```

Data Independence

- Physical data independence
 - The change in internal level does not have any effect on external level and conceptual level.
- Just like the previous one.
 - Except it is for internal level
- Changing file organization or storage structures should be possible without effecting the external level and conceptual level.

External Level

StudentID	First name	Last name	Tel	Address
-----------	------------	-----------	-----	---------

StudentID	AdvisorID	Advisor name
-----------	-----------	--------------

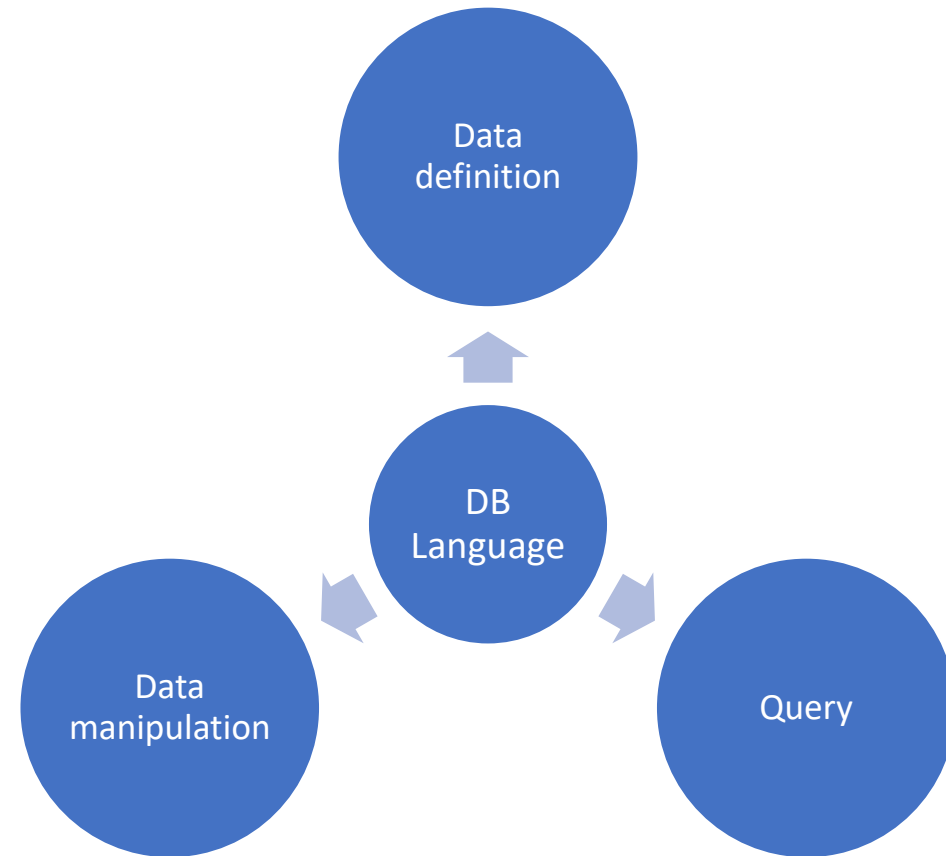
Conceptual Level

StudentID	First name	Last name	Tel	Address	AdvisorID	Advisor name
-----------	------------	-----------	-----	---------	-----------	--------------

Internal Level

```
public class Node{  
    int DataNumber;  
    String StudentID;  
    String Firstname;  
    String Lastname;  
    String Tel;  
    String Address;  
    String AdvisorID;  
    String AdvisorName;  
    Node next = new Node();  
    ...  
}
```

Database Language



Database Language

- Data definition language (DDL)

“A language that allows the DBA or user to describe and name the entities, attributes and relations required for the application, together with any associated integrity and security constraints.”

- The DDL is used to specify the definition of schemas.
- Not used to manage the data.

Database Language

- Data manipulation language (DML)

“A language that provides a set of operations to support the basic data manipulation operations on the data held in the database.”

- Provide the mechanism to
 - Insert new data
 - Update the existing data
 - Retrieve the data
 - Delete the data

Database Language

- Query language
 - A part in DML
- Used to retrieve the data from the database.

Relational Model and Language

Relational Model and Language

Objective of this lecture

- Explain the concept of the relational database.
- Identify the component of the database.
- Use the relational model in the system modeling

Relational Model and Language

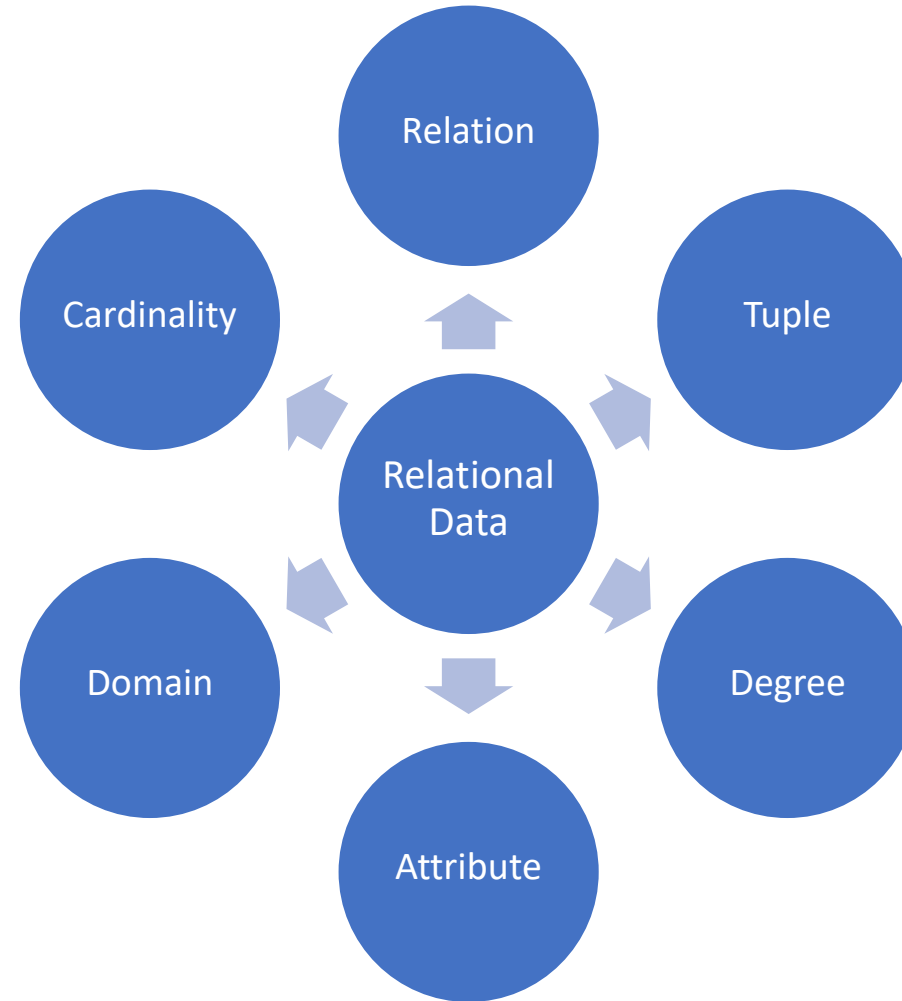
- Easy to understand
- One of the most popular
- Benefits
 - Data and program is not dependent.
 - Based on mathematical formulation
 - Set theory
 - Predicate logic

Relational data structure

Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

- What is column?
- What is similarity between the data in the same column?
- What is row?
- What is difference between the data in the different row?

Component of Relational data structure



Component of Relational data structure

- Relation *“A relation is a table with column and rows.”*
- Related to the external and conceptual levels of the database.

Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

Component of Relational data structure

- Attribute *“An attributed is a named column of a relation.”*
- The attribute is a place to hold the value about an object, entities.
- Relation = row and column
 - Row = record
 - Column = attributes

An attribute named student id



Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

Component of Relational data structure

- Domain *“Domain is a set of allowable values for one or more attributes”*
- Every attribute is defined by domain.
 - All of the value in an particular attribute is limited by domain.
- Different attribute may have different domain.
- Allows user to define a meaning of the attribute's value in one place.
 - If there is an incorrect data, it can be identified and handled.

A domain to describe a value of an citizen ID attribute.

Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

A domain to describe a value of an last name attribute.

Component of Relational data structure


- There are 3 types of domains.
- Data type
 - Int,float,string
- Range
 - Indicate the minimum acceptable value and maximum acceptable value
 - $0 < x < 10$
- Acceptable value
 - Specify the list of acceptable value.
 - {male,female}

Component of Relational data structure

- Tuple

“A tuple is a row of a relation.”

- Each tuple has values that conform with the attributes of the relation.
- The value of a tuple can be changed.
 - Someone updates the value.



Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

Component of Relational data structure

- Degree

“The degree of a relation is the number of attributes it contains.”

- Cardinality

“The cardinality of a relation is the number of tuples it contains.”

Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somying	Maisomwang	9911	222222222	Lampang

What is the cardinality?

What is the degree?

Notation for relation

$$R(A_1, A_2, \dots, A_n)$$

- R is a name of relation
- A1,A2,...,An are the name of the attribute
- Ex:

STUDENT (studentID,firstname,lastname)

Notation for tuple

$$tuple = \langle v_1, v_2, \dots v_3 \rangle$$

- v_1, v_2, \dots, v_n are the value of each corresponding attribute
- Ex:

STUDENT (studentID,firstname,lastname)

$\langle 520114586, "somsri", "hahahaha" \rangle$

Relational data structure

- Mathematical aspect of relations

- Given 2 sets

$$D_1 = \{1, 2\}$$

$$D_2 = \{A, B\}$$

- The Cartesian product of these 2 sets

$$D_1 \times D_2 = \{(1, A), (1, B), (2, A), (2, B)\}$$

Relational data structure

- Any subset of the Cartesian product is a relation.

$$R \subseteq D_1 \times D_2$$

$$R = \{(1, A), (2, A)\}$$

$$R = \{(1, A), (2, A), (2, B)\}$$

- Or, we can specify the member of the relation by setting some conditions. $R = \{(x, y) | x \in D_1 \wedge y \in D_2\}$

Relational data structure

- It can be extended beyond 2 domains.

$$R \subseteq D_1 \times D_2 \times \cdots \times D_n$$

Relational data structure

- Properties of relations
 - Ordering of tuples in a relation has no significance.
 - Since the tuple originates from set theory, there is no order in set.

$$\{1,2\} = \{2,1\}$$

- So,

Province	=	Province
Chiang Mai		Lamphun
Lamphun		Chiang Mai

Relational data structure

- Properties of relations
 - Ordering of attribute in a relation has no significance.
 - Similar to the order of tuple.
 - However, the order must be consistent with the defined order of attribute.

Province	Postal Code
Chiang Mai	50200
Lamphun	50100

Postal Code	Province
50200	Chiang Mai
51000	Lamphun

Relational data structure

- Properties of relations
 - Value in a tuple is atomic.
 - The value can not be divisible.
 - The empty value is NULL
 - No value
 - Unknown value
 - Provided value is not applicable

Relational data structure

- Properties of relations
 - No identical different rows in a relation.
 - According to the set theory, there are only a single copy a particular value in the set.

So,

$\{1\}$
 $\{1,1\}$ Not allows!!!!

Province	Postal Code
Ch...	
Mai...	
C...	
Ma...	ology

Relational data structure

- Properties of relations
 - Name of relation and name of attribute must be different.
 - The name of the relation in the same database must be unique.
 - No identical relation's name
 - The name of the attribute in the same relation must be unique.
 - No identical attribute's name

Relational data structure

- Properties of relations
 - Ordering of tuples in a relation has no significance.
 - Ordering of attribute in a relation has no significance.
 - Value in a tuple is unique.
 - No identical different rows in a relation.
 - Name of relation and name of attribute must be different.

Key

- Relation is a set of tuple.
- Each tuple is distinct.
 - No identical tuples.

Province	Postal Code
Chiang Mai	50200
Lamphun	50202
Chiang Mai	50202

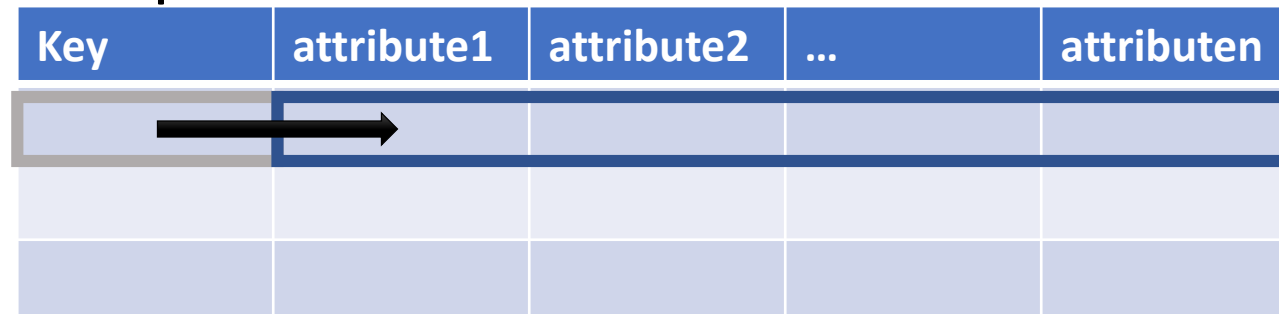
How to refer to “Chiang Mai at 50200”'s tuple

- A mechanism to refer to each tuple is needed!!!
- A unique attribute used to indicate each tuple is called **key**.

Key

- The key is a attribute/ set of attributes that can be used to **uniquely** identified each tuple.

Key	attribute1	attribute2	...	attributen

A diagram illustrating the concept of a key in a database. It shows a table with five columns: 'Key', 'attribute1', 'attribute2', '...', and 'attributen'. The first row of data is highlighted with a thick blue border. A black arrow points from the 'Key' cell in this row to the 'attribute1' cell, indicating that the key is used to uniquely identify each tuple.

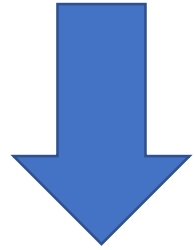
- The key can be used to point to a certain particular tuple without ambiguity.



attribute1	attribute2	...	attributen
1	Z		00001
1	Y		00002
2	Z		00001

1 can point to 2 tuples.

Attribute2,...,attribute n have the similar problem.



Key	attribute1	attribute2	...	attributen
A	1	Z		00001
B	1	Y		00002
C	2	Z		00001

A can only point to a certain tuple.
B can only point to a certain tuple.
C can only point to a certain tuple.

It is a KEY !!!!

Key

- There are 4 types of key.
 - Primary key
 - Foreign key
 - Candidate key
 - Superkey

Key

- Superkey *“An attribute, or set of attributes that uniquely identifies a tuple within a relation”*

Student ID	First name	Last name	Tel	Citizen ID	Address
590551020	Somchai	Somwang	1199	111111111	Lamphun
590551021	Somchai	Maisomwang	9911	222222222	Lampang

- Every relation must have a “default superkey”.
 - Every attribute of the relation constitutes the default superkey.

Key

- Candidate key
“A superkey such that no proper subset is a superkey within the relation.”
- The candidate key, K , must exhibit :
 - Uniqueness
 - The candidate key, K , must be unique.
 - No repeated
 - Irreducibility
 - No proper subset of K is unique.
- There can be more than one attribute in the candidate key.

Student ID	Course ID	Semester	grade
1001	113	1/58	A
1002	114	2/58	F
1001	212	1/59	B
1002	113	2/58	A
1003	212	2/58	D
1002	114	1/59	D+
1001	114	1/58	A
1002	212	1/59	B

{student ID}, {course ID}, {semester}, {grade}

{student ID, course ID}, {student ID, semester}, {student ID, grade},

{course ID, semester}, {course ID, grade}, {semester, grade}

{student ID, course ID, semester}, {course ID, semester, grade},

{student ID, course ID, grade}, {student ID, semester, grade}

{student ID, course ID , semester, grade}

What is wrong?

Set theory : Proper Subset

- Definition

*“A proper subset of A is a subset of A that is **not equal** to A*

- Example

$$A = \{a, b, c\}$$

$$B = \{a, b\} \quad B \text{ is a proper subset of } A.$$

$$C = \{b, c\} \quad C \text{ is a proper subset of } A.$$

$$D = \{a, b, c\} \quad D \text{ **is not** a proper subset of } A.$$

Student ID	Course ID	Semester	grade
1001	113	1/58	A
1002	114	2/58	F
1001	212	1/59	B
1002	113	2/58	A
1003	212	2/58	D
1002	114	1/59	D+
1001	114	1/58	A
1002	212	1/59	B

{student ID}, {course ID}, {semester}, {grade}

{student ID, course ID}, {student ID, semester}, {student ID, grade},

{course ID, semester}, {course ID, grade}, {semester, grade}

{student ID, course ID, semester}, {course ID, semester, grade},

{student ID, course ID, grade}, {student ID, semester, grade}

{course ID, semester, grade}

Key

- When a candidate key consists of more than one attribute, we call it composite key.

Key

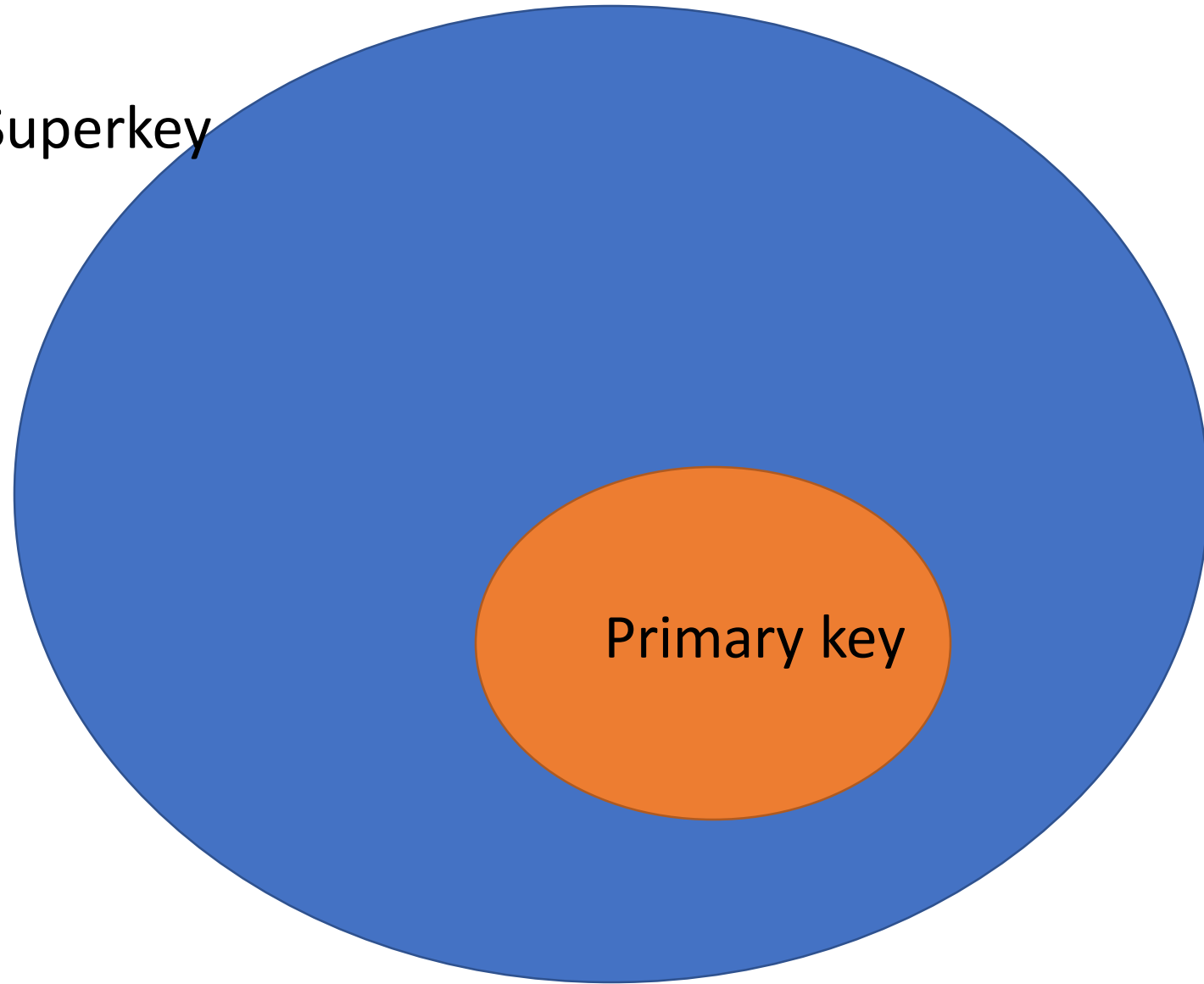
- Primary key
"The key that is selected to identify tuples uniquely within the
- The primary key must exhibit :
 - Uniqueness
 - The primary key must be unique.
 - No repeated
 - Minimality
 - The primary key must consist of the smallest number of attribute to uniquely identify a tuple.

Student ID	First name	Last name	Tel	Address
590551020	Somchai	Somwang	1199	Lamphun
590551021	Somchai	Maisomwang	9911	Lampang
591325465	Somsri	Somwang	1199	Lamphun

Which tuple can be used as a primary key?

Which tuple can be used as a candidate key?

Superkey



Primary key

Key

- Foreign key

“An attribute or set of attributes within one relation that matches the candidate key of some (possibly the same) relation.”

- It is used as a reference to other relation.

Key

- In the relation, the key can be indicated by underlining the attribute.

STUDENT (studentID,firstname,lastname)

STUDENT (studentID,firstname,lastname)

Warning !!!!

- You have to choose the key carefully.
- Some attribute can only be used as a key for a certain period of time.
 - When the data grows bigger, it might not be able to use as a key.
- We tend to select the key with smallest number of attribute.

Integrity Constraints

- The data model consists of 2 parts.:
 - Data
 - Description of data
- The description of data provides :
 - List of operation that can perform on the data
 - The **integrity constraint** to ensure the accuracy of the data

Integrity Constraints

- Given the following table,

Student ID	First name	Last name	Tel	Address
590551020	Somchai	Somwang	1199	Lamphun
590551021	Somchai	Maisomwang	9911	Lampang
591325465	Somsri	Somwang	1199	Lamphun

- What if I accidentally put the tel in the student ID or first name?

Integrity Constraints

- There are 3 types of integrity constraints.
 - Domain constraint
 - Entity constraint
 - Referential integrity

Integrity Constraints

- Domain constraint

“Each value of a certain attribute must be in the corresponding

- This rule is to ensure the accuracy of data.

Integrity Constraints

- Entity integrity

“In a base relation, no attribute of a primary key can be null.”

- This rule is to each tuple is unique and is able to refer.

Integrity Constraints

- Referential integrity
“If a foreign key exists in a relation, either the foreign key value must match a candidate key value of some tuple in its home relation or the foreign key value must be wholly null.”
- This one is for multiple relation.
- This rule is to ensure the consistency of the referring.
- If the relation refers to other relation through foreign key, there must be data that the tuple points to.