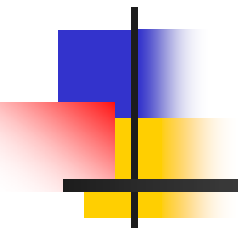
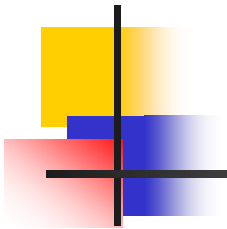


Relational Database Design:

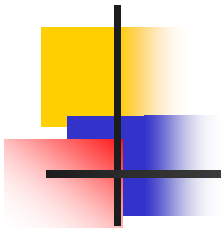
Recap of ICS 334

The first DB course in KFUPM



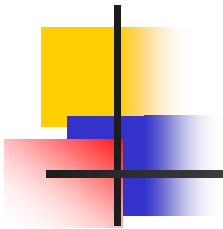
Announcement

- Read the following paper before next class:
- Code, *Relational Model of Data Large Shared Data Banks*, communication of the ACM, Volume 12, number 6, June 1970.
  - Posted for you in the WebCT.



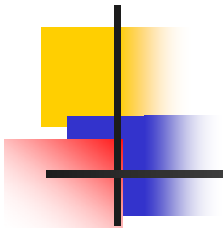
Lecture objectives

- Briefly mention topics covered in ICS 334.



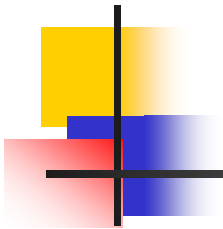
- Lecture outline

- The main steps in DB Design
- Requirement collection and analysis
- Conceptual data model
- Logical data model
- Physical data model
- Relational Data model



- The main Steps in DB Design

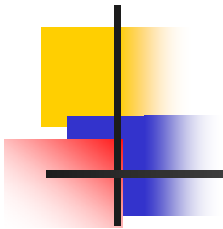
- Requirement collection and analysis
  - How the data is collected and analyzed.
- Data modeling
  - How data should be stored in the database.
  - Includes:
    - Conceptual data model
    - Logical data model
    - Relational data model
    - physical data model
- Functional modeling
  - How the data is processed.



- Requirement collection and analysis

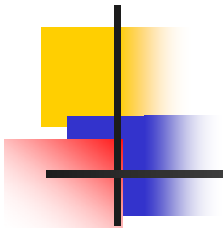
- The goals are:
  - To determine the data requirements of the DB in terms of primitive objects
  - To classify and describe the information about these objects
  - To identify and classify the relationships among the objects
  - To determine the types of transactions that will be executed on the DB and the interactions between the data and the transactions
  - To identify rules governing the integrity of the data

- Requirement analysis can be gathered by data modeler from:
  - Existing documents
  - Users
  - Existing systems



-- Example

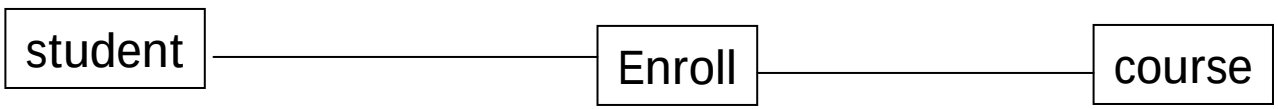
- Required: A database to record students' enrollment in ICS courses.
- Assume that after requirement analysis we identified the following 3 entities.
  - Course
  - Enroll
  - Student



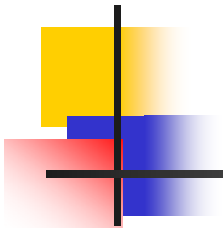
- Conceptual data model

- Objectives:
  - To identify entities:
  - To identify the highest-level relationships among entities.
- No attribute is specified

-- Example:



3 main entities: Student, Enroll, Course

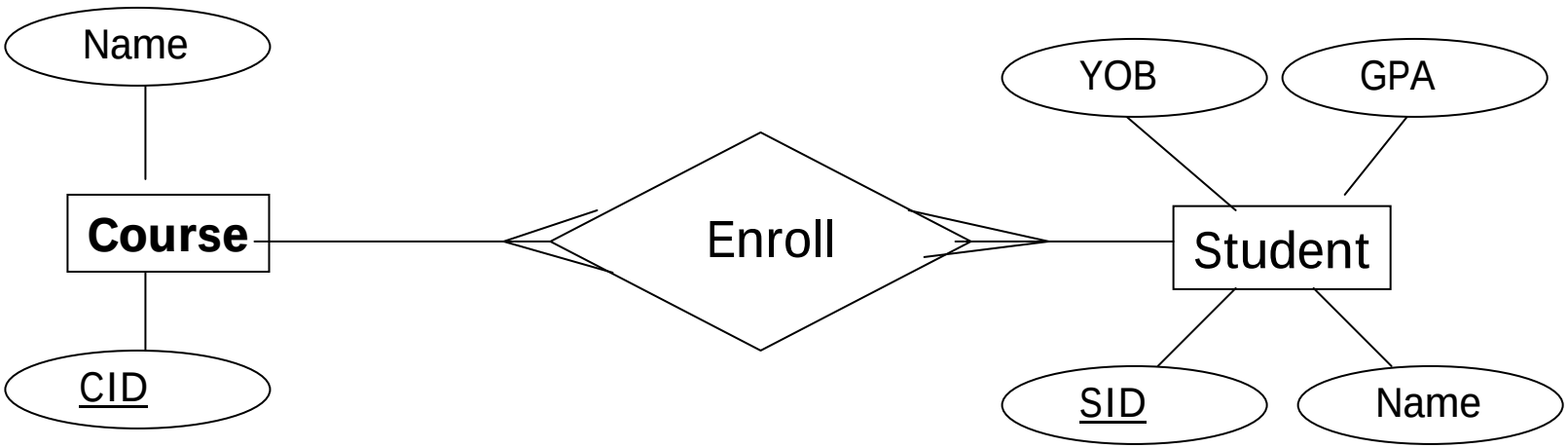


- Logical data model

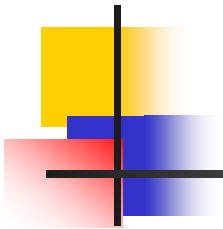
- Objective:
  - Describe the data in as much detail as possible, without regard to how they will be physically implemented in the database.

- Features include:
  - Entities and relationships among them.
  - All attributes for each entity are specified.
  - The primary key for each entity specified.
  - Foreign keys are specified.
  - Many-to-many relationships are resolved
  - Normalization occurs at this level.

-- Example:

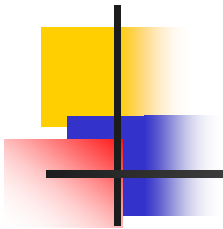


Note: Because relational data model is a logical data model, it should be discussed together with the logical data model. But I prefer discussing it after the physical data model.



- Physical Data Model

- Features:
  - Specification all tables and columns
  - Foreign keys are used to identify relationships between tables
  - Denormalization may occur based on user requirements
  - Physical considerations may cause the physical data model to be quite different from the logical data model
- Steps:
  - Convert entities into tables
  - Convert relationships into foreign keys
  - Convert attributes into columns
  - Modify the physical data model based on physical constraints / requirements



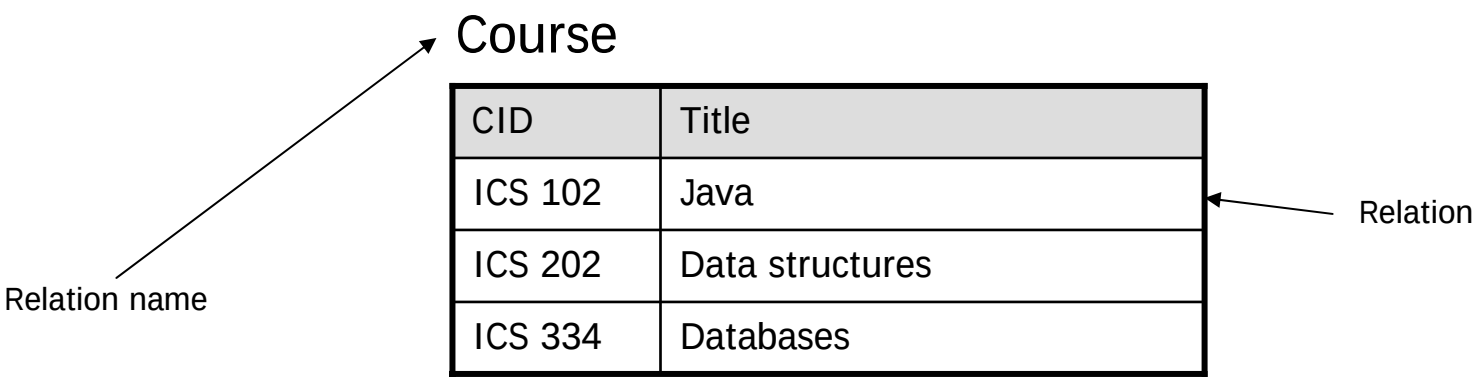
- The Relational Data Model

- The logical model behind the relational DB (RDB). Data is always represented as relations (2-dim. tables).
- Basic Concepts:
  - Relation
  - Attribute
  - Schema
  - Tuple
  - Keys
  - Constraints
  - Functional dependency
  - Normalization

-- Relation

- The way to represent data is through relations.
- A relation is a two-dimensional table.
- The order of rows and columns can be exchanged, and it is still the same relation.

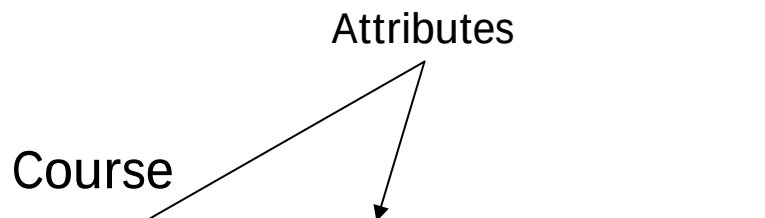
Example:



-- Attribute

- An attribute is the name of a column in a relation. It usually describes the meaning of the content in the column.

- Example:



CID	Title
ICS 102	Java
ICS 202	Data structures
ICS 334	Databases

-- Schema

- A schema is a description of a class of relation. It consists of the name of the relation and the set of attributes in the relation.

  That it is a set of attributes means that the attributes are unordered.

- Example:

Course

CID	Title
ICS 102	Java
ICS 202	Data structures
ICS 334	Databases

Schema for the above relation: Course(CID, Title)

-- Tuple

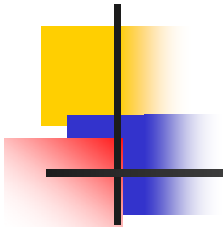
- A tuple is a row in a table. A relation can be seen as a set of tuples.

- Example:

Course

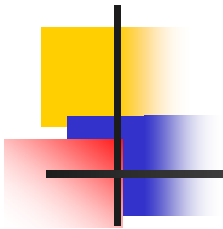
CID	Title
ICS 102	Java
ICS 202	Data structures
ICS 334	Databases

A relation with 3 tuples



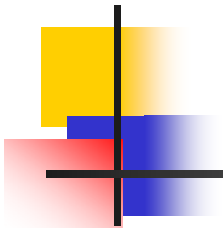
-- Keys

- Superkey:
  - A combination of attributes that can be uniquely used to identify a record.
  - A table may have many superkeys
- Candidate key:
  - A superkey but without extraneous data.
  - A table can have one or more candidate keys
- Primary key
  - One of the candidate keys is chosen to be the primary key.
  - There can only be one primary key in a table
- Alternate key
  - All the candidate keys, minus the primary key.
  - The number of alternate keys in a table is one less than the number of candidate keys.



-- constraints

- NOT NULL
  - No null values are allowed in an attribute specified as NOT NULL
- UNIQUE
  - No duplicate values are allowed in a column specified as UNIQUE.
- Primary key
  - Must be UNIQUE and NOT NULL
- Foreign key
  - Must refer to a value of a candidate key in the parent table
  - Can be null
  - Can be duplicate



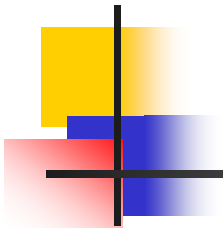
-- Functional dependency

- A functional dependency occurs when one or more attributes in a relation uniquely determine other attribute(s). If A and B are attributes, this can be written A --> B which would be the same as stating "B is functionally dependent upon A."

- Attribute A is the **determinant** and attribute B is the **functionally dependent.**

- If the determinant is part of the primary key (and not the whole key), then the dependency is called **partial dependency**.

- If both the determinant and the functionally dependent attributes are non key attributes, then the dependency is called **transitive dependency.**



-- Normalization

- UNF (Un Normalized form)
  - A relation is in UNF if it contains at least one multi-valued attribute
- INF
  - A relation is in 1NF if it has no multi-valued attribute.
- 2NF
  - A relations is in 2NF if it is in 1NF and has no partial dependency.
- 3NF
  - A relation is in 3NF if its is in 2NF and contains no transitive dependency.
- BCNF
  - A relation is in BCNF if and only if, every determinant is a candidate key.

END