

Lecture 3a

Data Models and Database Schemas

Week 2

Overview

- What is a data model
- Database schema and database instance

Data model

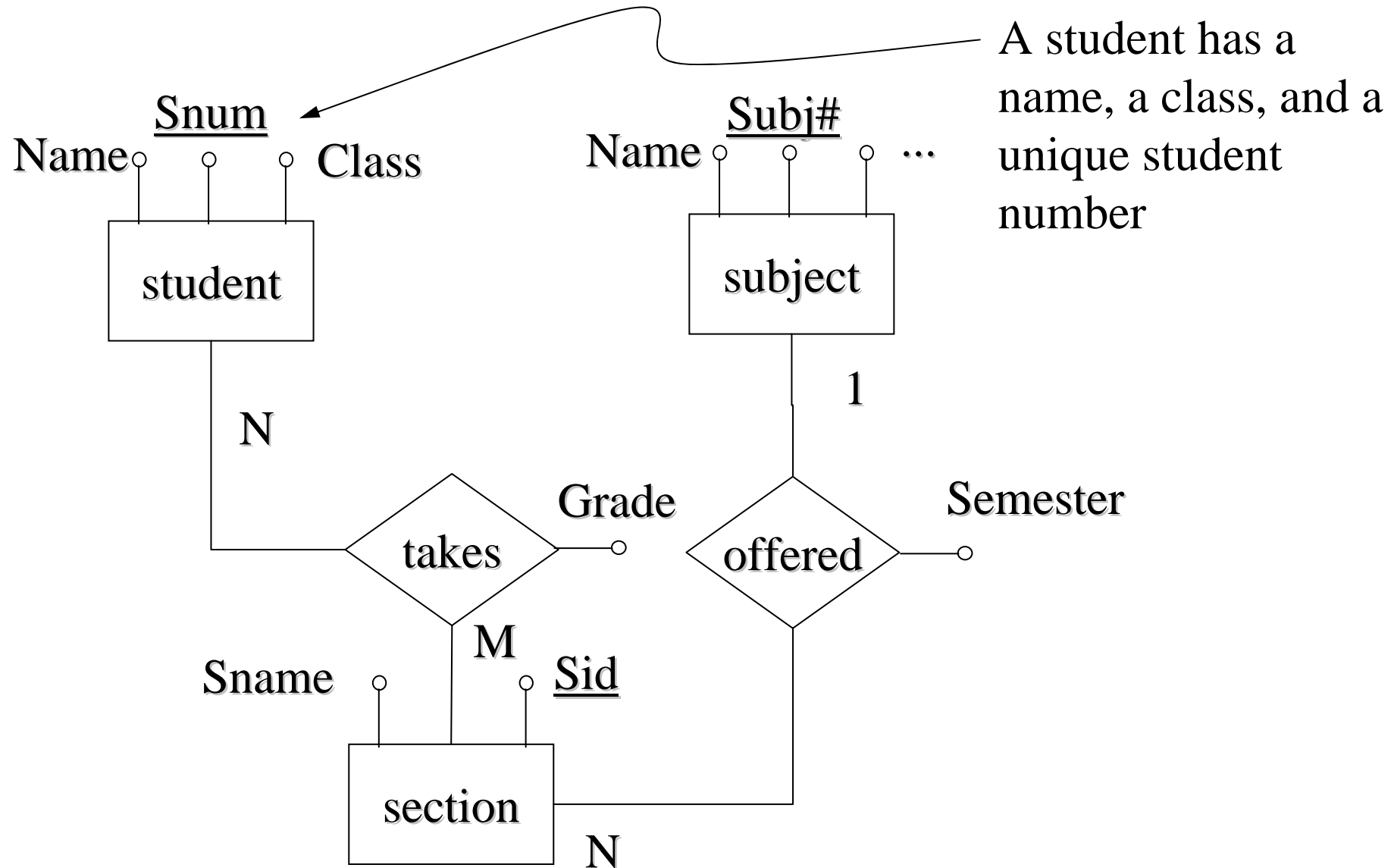
- A set of generic concepts suitable for the *description* of data, relationships among- and constraints on data
- Typically a data model also defines useful generic *operations* on the data

Levels of abstraction in data models

- We have various *levels* of data models. The level depends on how close are the data model's concepts to the concepts utilised in the Universe of Discourse

- Conceptual level (close to end user, can be used for *communication between designers and users*)
 - Extended Entity Relationship Model (EER, IDEF1X, ...)
 - Object-Oriented Data Model (UML, EXPRESS,...)
- Logical level (*closer to implementation*, simple, record based)
 - relational data model
 - network data model
 - hierarchical data model
- Physical level (describes physical storage / *implementation details*, e.g. index files, etc.)

Example: a conceptual level database schema

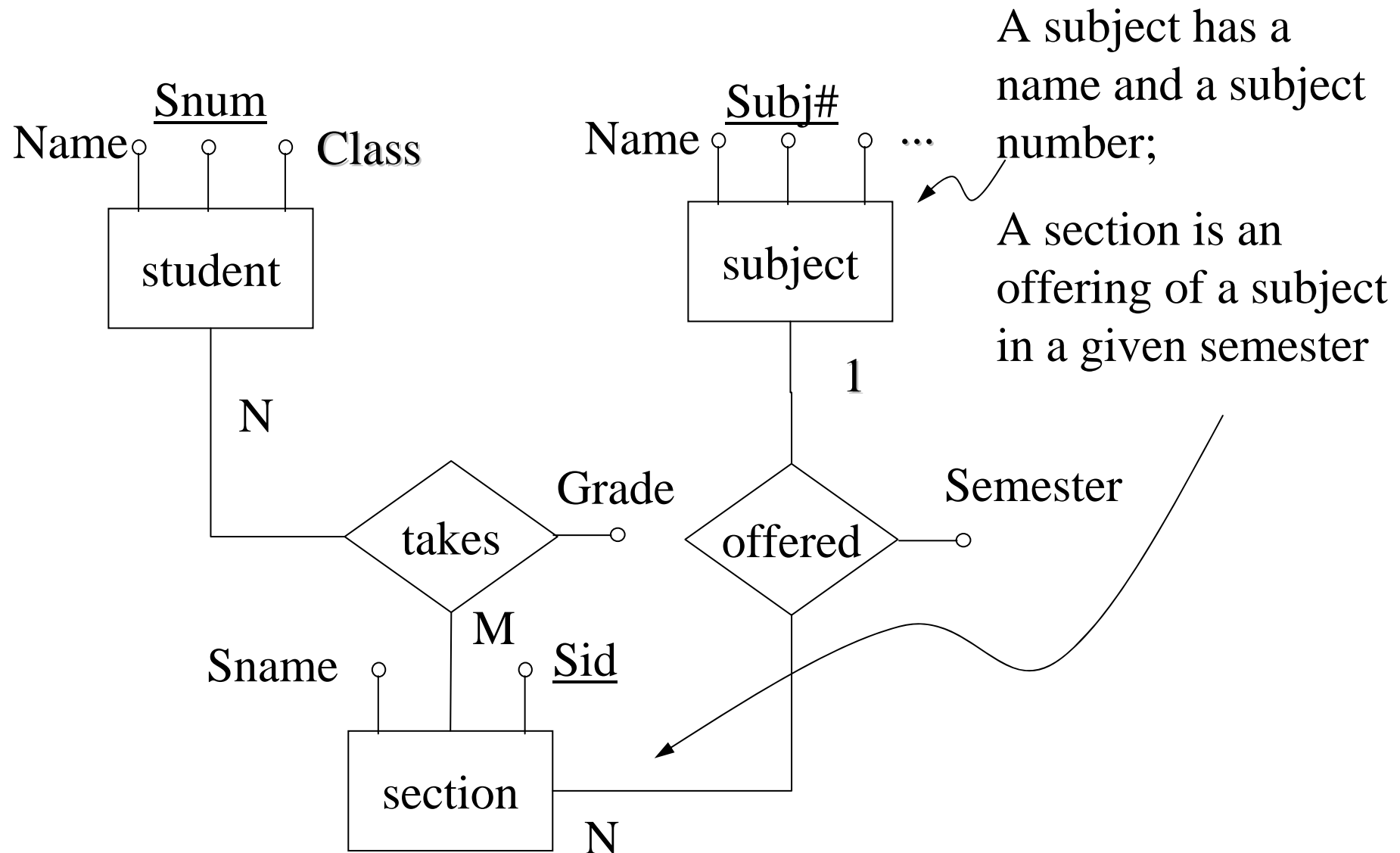


The basic concepts of the Entity Relationship data model (as one of the conceptual level data models)

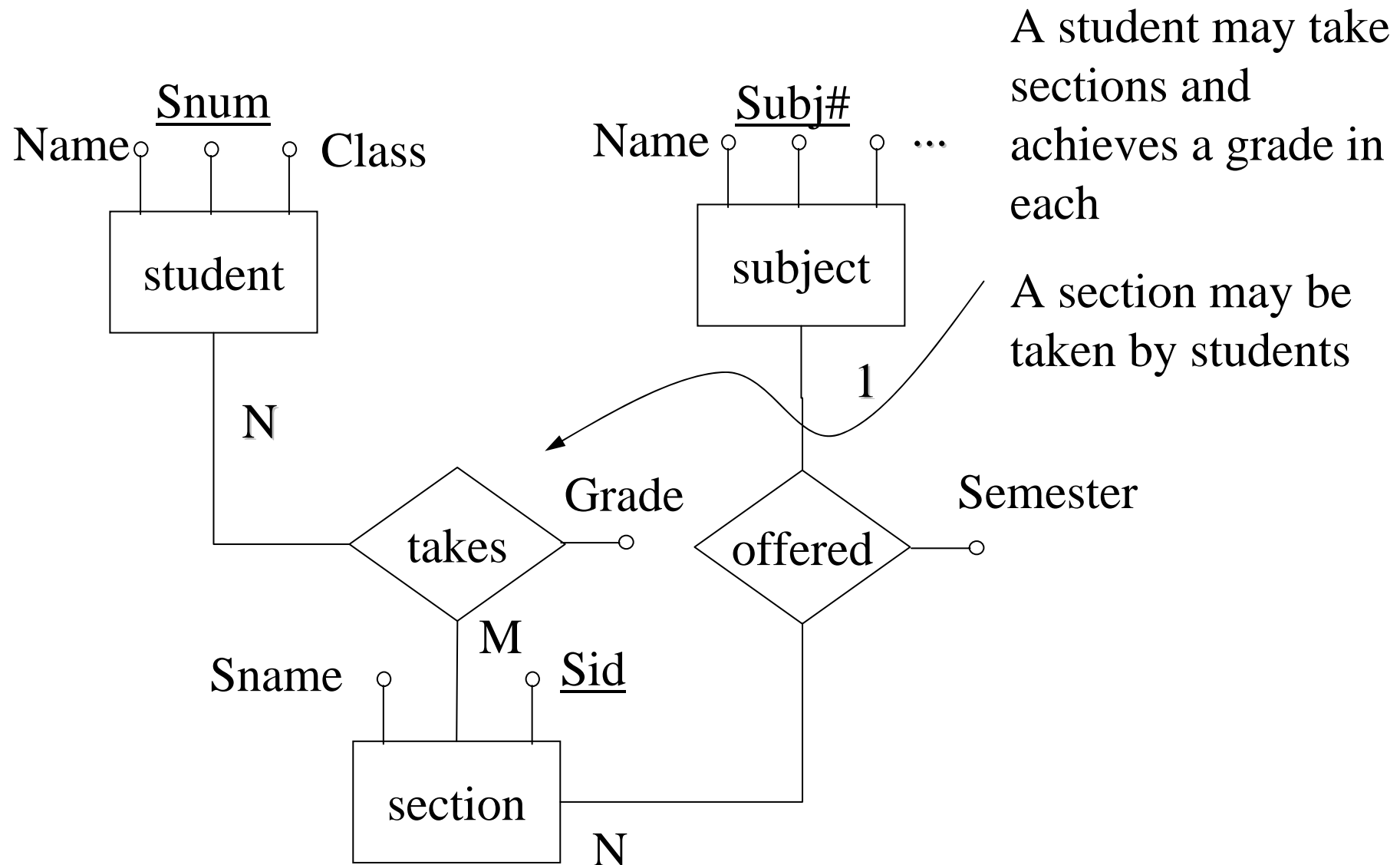
- *Entity* (e.g. Student, Subject,...)
- *Relationship* (e.g. Takes, Offered,...)
- *Attribute* (e.g. Name, Semester)

(there are some auxiliary concepts as well - to express constraints)

Example: conceptual level



Example: conceptual level



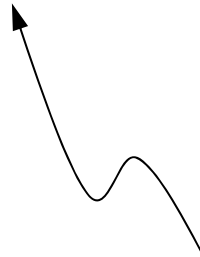
Example: a logical level database schema

STUDENT (Name, Snum, Class)

SUBJECT (Subj#, Name,...)

SECTION (Sname, Sid, Subj#)

STUDENT TAKES (Snum, Sid, Grade)



Data storage can be thought of as if data were stored in tables
(e.g. the above four tables)

The name “logical” refers to the possibility of storing data as
a collection of logical propositions

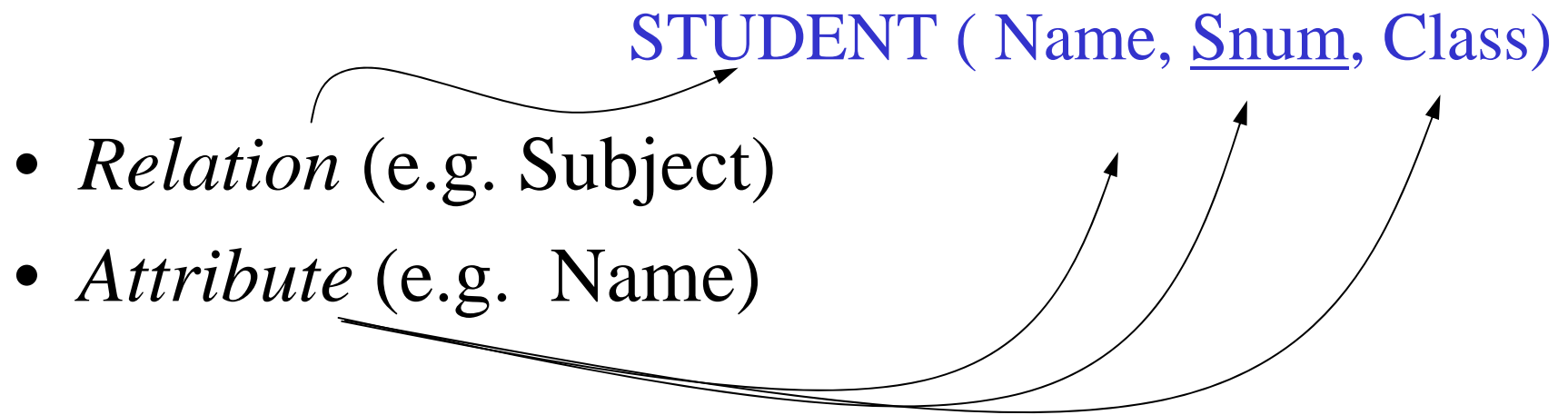


NOTE

In the Database literature (especially in manuals of CASE tools) the conceptual level is sometimes called the *logical* level, and the logical level is called the *physical* level. As a consequence they have no separate name for the physical level.

This is unfortunate, but we can live with it...

The basic concepts of the Relational data model (as one of the logical level data models)



(again, there are some auxiliary concepts as well, to express certain constraints)

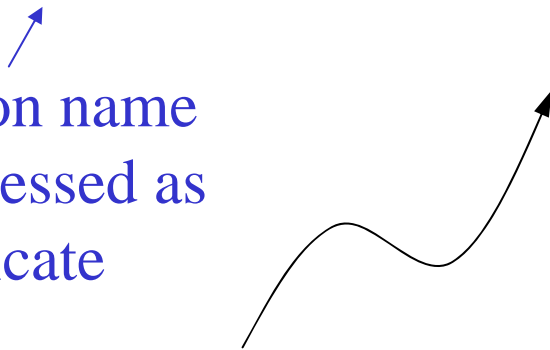
Each relation is represented as a set of propositions:

STUDENT (M.Smith, 123456, CIT5005).


STUDENT (P.Smith, 234561, CIT5005).

STUDENT (W.Smith, 345612, CIT5005).

Relation name
is expressed as
a predicate



Attributes are
atomic values
(numbers, strings)



A *relation* is a set of propositions with the same predicate and ‘arity’ (e.g. in this example the predicate is ‘STUDENT’ and the ‘arity’ is 3 because the relation has three attributes).

Each proposition is called a *relationship*. Here we have three propositions.

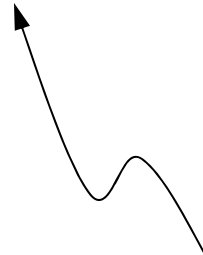
Example: physical level database schema

Logical level schema + storage characteristics

INDEX (“STUDENT”, “Snum”, “StudentIndex”)

DBFILE (“STUDENT”, “/home/DBMS/Dbfiles/STUDENT”)

.....



Data storage characteristics - index files, etc are defined in a proprietary language which may be an extension of a standard DDL/DML, such as SQL.

The basic concepts of physical data model(s)

- Physical data models use the concepts of the logical level data model

+

- Any concepts necessary to describe physical storage characteristics (*index, file, ...*)

Database schema

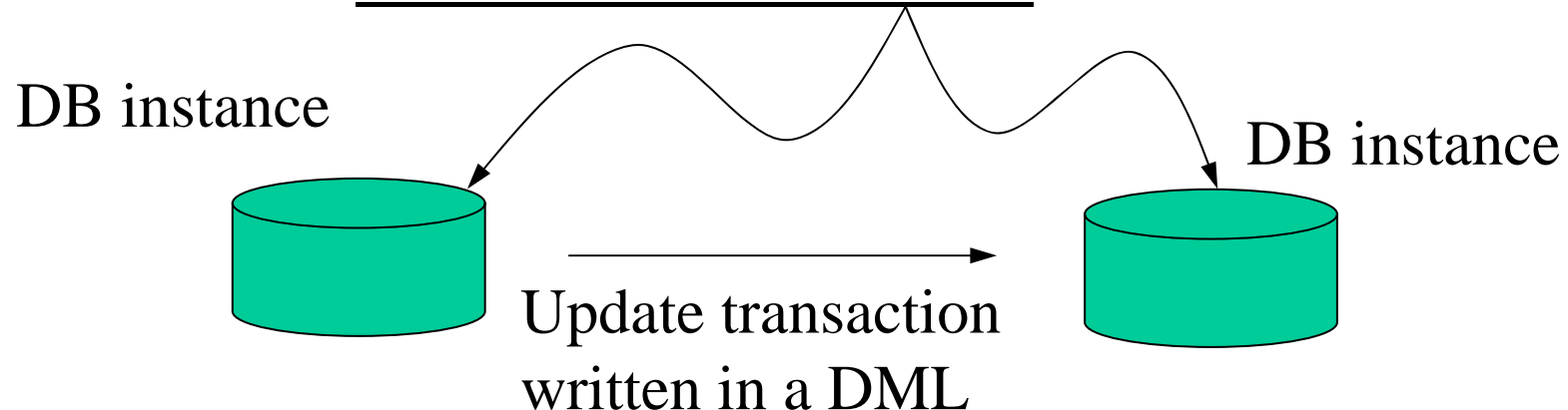
- A database schema is a description of the structure of data to be stored in the database
(E.g. in the relational data model it describes the names of relations, attributes, and constraints)
- A database schema is defined using a Data Description Language (DDL)
- The description is stored in a Data Dictionary ('catalog') which, by the way, is one of the files managed by the Database Management System itself

Database instance

- A database instance is the collection of data at any moment in time
- Many database instances correspond to one schema
 - the database instance continuously changes as a result of *insert*, *update* and *delete* transactions
 - the database schema is usually very stable and changes infrequently

Database instance (cont'd)

- We use a Data Manipulation Language (DML) to transform database instances from one into another.
- The DBMS ensures that each database instance satisfies the schema



The end