Database Design

Marta Jakubowska-Sobczak
IT/ADC

based on slides prepared by
Paula Figueiredo, IT/DB

Outline

- Database concepts
- Conceptual Design
- Logical Design
- Communicating with the RDBMS

Some concepts

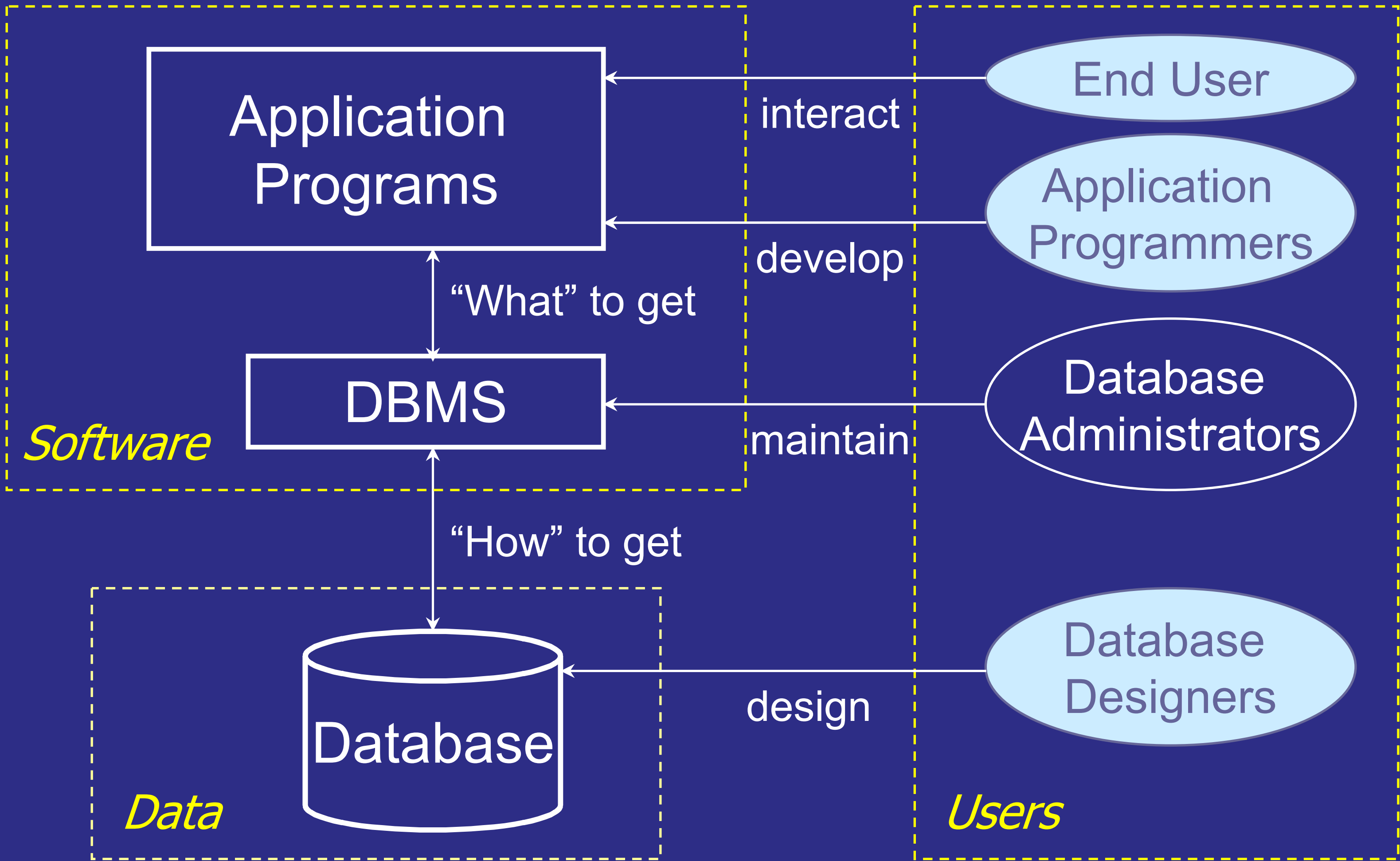
- **Database:** an integrated collection of logically related data.
- **Data:** known facts that can be stored (text, graphics, images,...).
- **Database management system (DBMS):** system that stores, retrieves, and modifies data in the database.

Some concepts

- **Database schema:** a collection of database objects.

- **Relational database:** Collection of two-dimensional tables.

Database systems



Why is design important?

- Why do I need a database?
  - To store data…
  - …and be able to process it to produce information!
- Define how to store the data.

Design goals

- Store information without unnecessary redundancy
- Retrieve information easily and accurately
- Be scalable for data and interfaces

Database Design Phases

- **Conceptual Design**
  - Produces the initial model of the real world in a conceptual model
- **Logical Design**
  - Consists of transforming this schema into the data model supported by the DBMS
- **Physical Design** (not covered)
  - Aims at improving the performance of the final system

Conceptual Design

- Process of constructing a model of the information used in an enterprise.
- Is a conceptual representation of the data structures.
- Is independent of all physical considerations.

Conceptual Design

- **Input:** database requirements
- **Output:** conceptual model

- It should be simple enough to communicate with the end user
- It should be detailed enough to create the physical structure

Entity Relationship Model

- The Entity-Relationship model (ER) is most common conceptual model for database design
- Describes the data in a system and how data is related
- Describes data as **entities**, **attributes**, and **relationships**

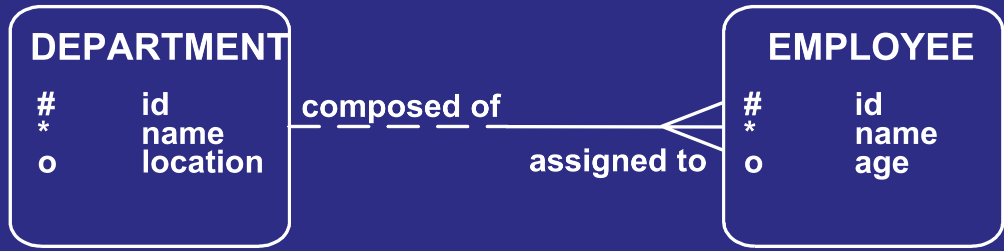
Entities

- Concepts (abstract or real) about which information is collected
  - Ex: employee, region, building, project, registration
- Entity Instance: entity individual occurrence

Attributes

- Properties which describe the entities
  - Ex: Name (attribute of entity employee)
- Identify, describe, classify or quantify an entity
- An attribute instance is a *value*
  - "John Smith" is a value of attribute Name
- Characteristics: optionality
  - Required or optional

Relationship



- Associations between entities
  - Ex: employees *are assigned to* projects

    departments *manage* projects

Degree: number of entities associated with a relationship (most common are binary)

Cardinality: indicates the maximum possible number of entity occurrences

Existence: indicates the minimum number of entity occurrences

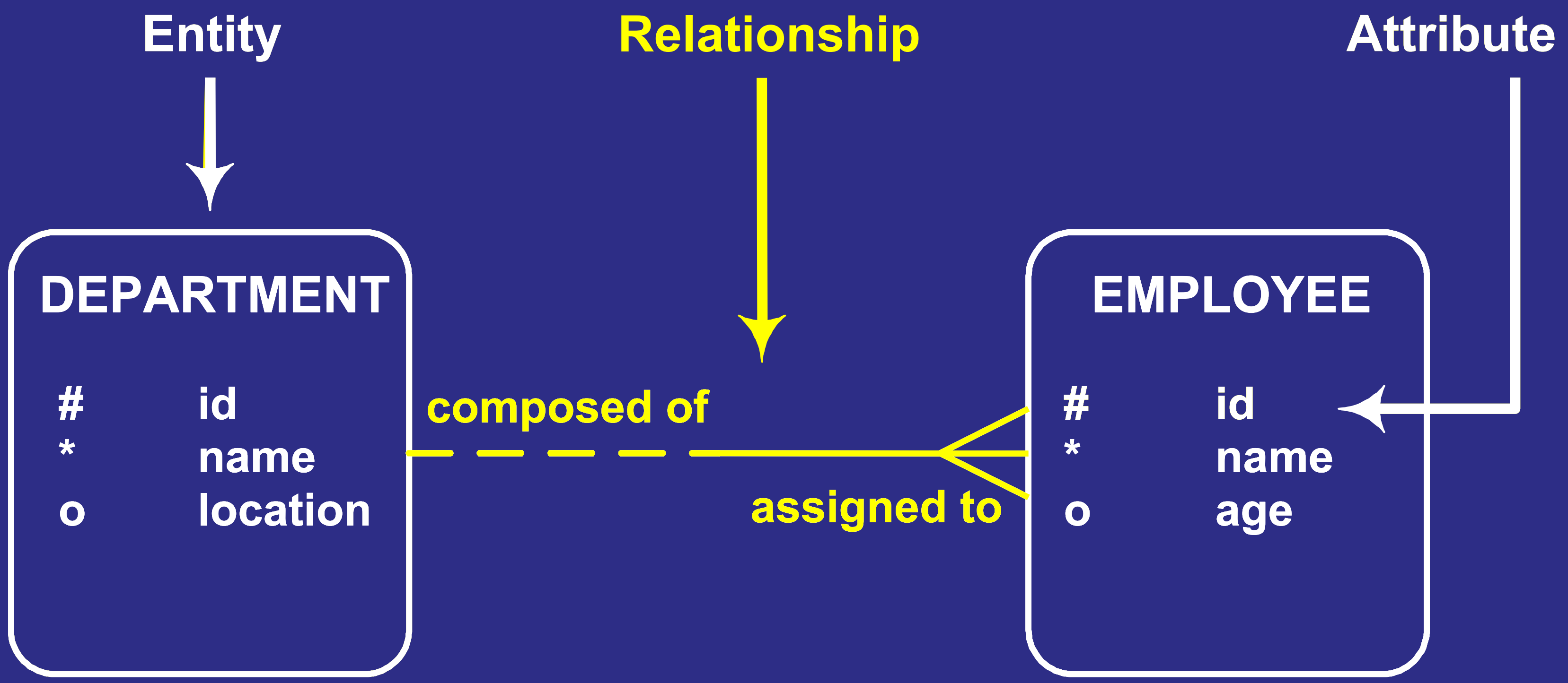
Relationship cardinality

- one-to-one (1:1)
  - A manager is head of a department
- one-to-many (1:N)
  - A department has many employees
  - Each employee is assigned to one department
- many-to-many (M:N)
  - Employees are assigned to many projects
  - Projects have assigned several employees

Relationship existence

- Mandatory (must be)
  - Every project must be managed by a department
- Optional (may be)
  - Employees may be assigned to work on projects

ER Diagram Notation

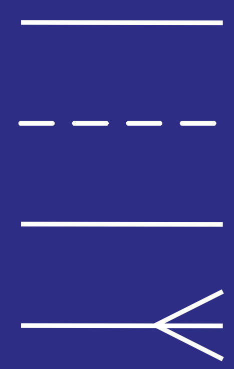


ER Modelling conventions

- Entity
  - Soft box
  - Singular name
  - Unique
  - Uppercase
- Attribute
  - Singular name
  - Unique within the entity
  - Lowercase
  - Mandatory (*)
  - Optional (o)
  - Unique identifier (#)

ER Modelling conventions

- Relationship
  - Line connecting to entities
  - Name: descriptive phrase (assigned to, composed of)
  - Mandatory: solid line
  - Optional: dashed line
  - One: single line
  - Many: crow's foot

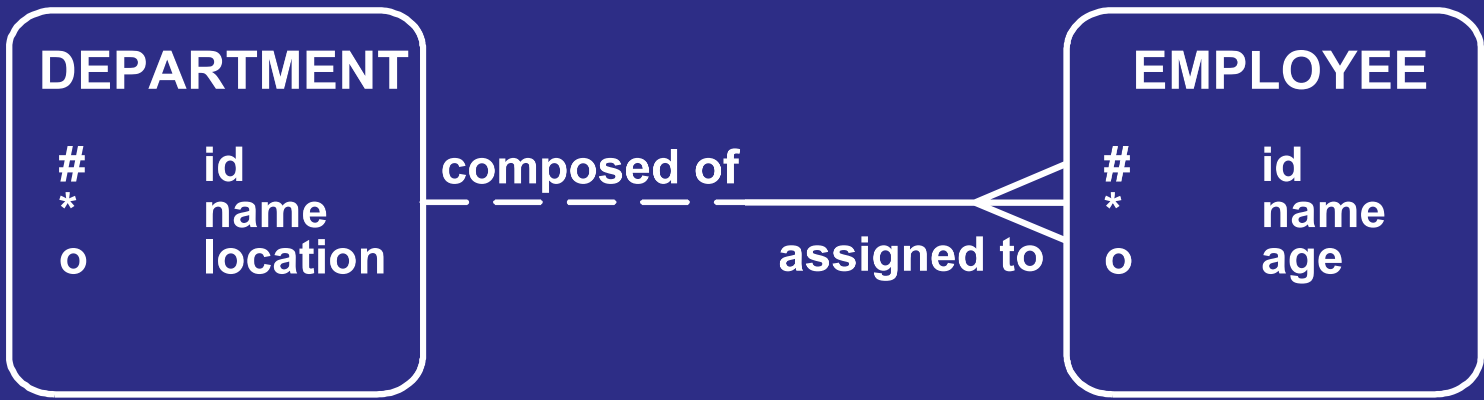


ER Notation

- Are read clockwise

- Each source entity {may be| must be} relationship name {one and only one| one or more} destination entity

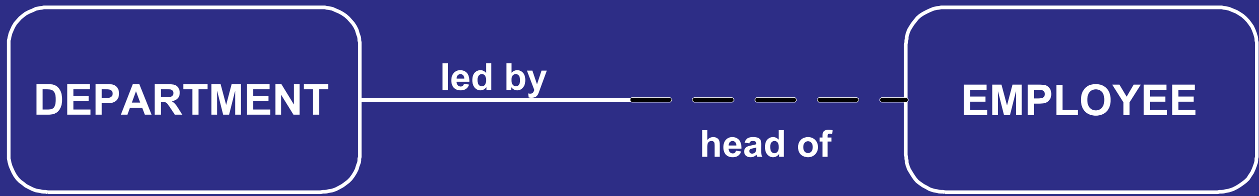
ER Notation



- Each DEPARTMENT may be composed of one or more EMPLOYEES
- Each EMPLOYEE must be assigned to one and only one DEPARTMENT

Relationships examples

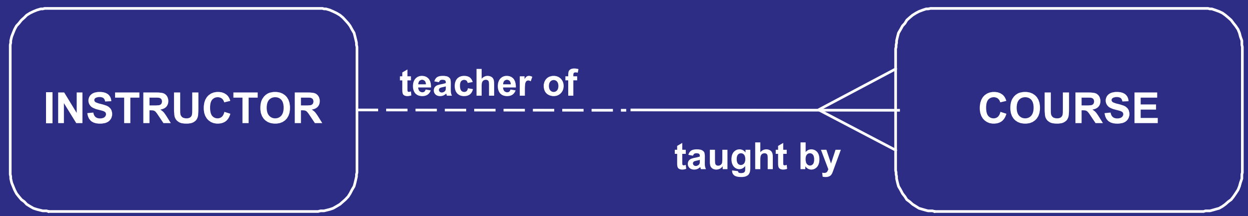
- One-to-one (1:1)



  - Every department is led by exactly one employee
  - An employee can be head of at most one department

Relationships examples

- One-to-many (1:N)



  - Each instructor may be teacher of one or more courses
  - Each course must be taught by one and only one instructor

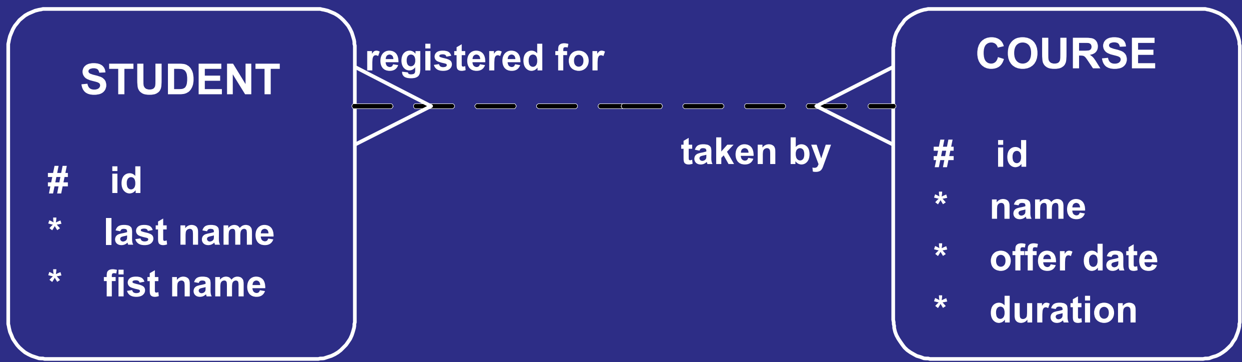
Relationships examples

- Many-to-many (M:N)

  - A student can be registered on any number of courses (including zero)
  - A course can be taken by any number of students (including zero)

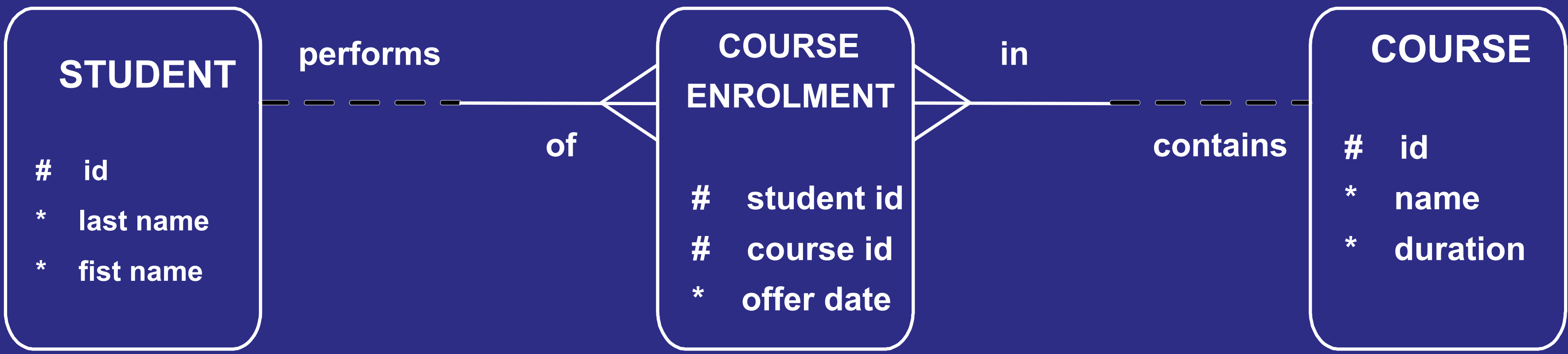
Modelling Relationships

- Many-to-many
  - Cannot be represented by the relational model
  - Solution: create intersection or associative entities



Modelling Relationships

- Many-to-many



  - A student may perform an enrolment in a course
  - A course may contain enrolments of students

Logical Design

- Translate the conceptual representation into the logical structure
- **Input**: conceptual model (ERD)
- **Output**: relational model, normalized relations

Relational Model

- Represents data in the form of relations
- Data structure
  - data stored in the form of relational tables
- Data integrity
  - Tables have to satisfy integrity constraints
- Data manipulation
  - Operations are used to manipulate the stored tables to generate information

Data structure: table

- Composed by named columns and unnamed rows
- The rows represent occurrences of the entity

EMP(empno, ename, job)

Attribute: column

Cell/ Field

Tuple:row/record

EMPNO	ENAME	JOB
7369	SMITH	CLERK
7499	ALLEN	SALESMAN
7654	MARTIN	SALESMAN
7566	JONES	MANAGER
7788	SCOTT	ANALYST
7844	TURNER	SALESMAN

Relational table characteristics

- Every table has a unique name
- Columns in tables have unique names
- Every row is unique
- Order of rows is irrelevant
- Order of columns is irrelevant
- Every field value is atomic (contains a single value)

Primary keys (PK)

- Attribute or set of attributes that uniquely identify an entity instance
- Role: Enforce integrity
- Every entity in the data model must have a primary key

Identifying Primary keys

- For each entity instance, an attribute
  - must have a non-null value
  - the value must be unique
  - the values must not change or become null during the table life time (time invariant)
- Attributes with this characteristics are <u>candidate keys</u>.

Foreign keys (FK)

- An attribute in a table that serves as primary key of another table
- Enforce referential integrity by completing an association between two entities

Data integrity

- Why?

  To Avoid data redundancy and inconsistency

- How?

  Using integrity constraints rules

Integrity constraints

- Domain Constraints
  - All values of an attribute should be taken from the same domain.
  - How to define a domain?

Attribute	Domain Name	Description	Domain
Price	prices	Set of all possible book prices	Monetary 6 digits

Integrity constraints

- Entity integrity
  - Every table must have a primary key.
  - Primary key must be not-null for each entity instance

  Insert, update and delete operations must maintain the uniqueness and existence of all primary keys.

Integrity constraints

- Referential integrity
  - Maintains consistency between two tables with a relation.
  - The foreign key must
    - have a value that matches a primary key in the other table.
    - Or be null.

Referential integrity

EMP (empno, ename, job, projno, deptno)

PROJ(projno, pname, loc)

DEPT(deptno, dname, loc)

- An employee has to belong to a department:
  FK deptno can not take null value (mandatory).
- An employee may be assigned to a project:
  FK projno can take null value (optional).
- On both cases the FK must have a value that match a PK value in the other table.

Referential integrity

Primary key ↓ | Foreign keys ↓ ↓

EMPNO	ENAME	JOB	PROJNO	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKE	MANAGER		30
7782	CLARK	MANAGER	102	10
7566	JONES	MANAGER	100	20

PROJNO	PNAME	LOC
100	SMNG	BOSTON
102	STR	SEATLE

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

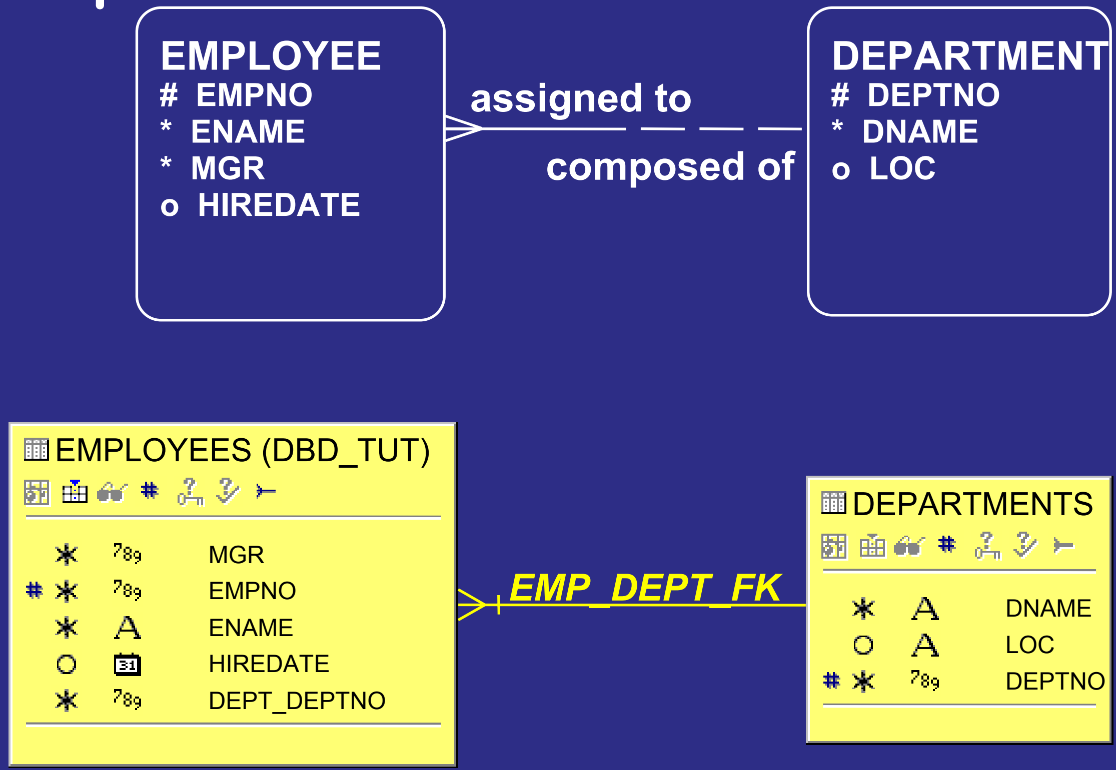
From ERD to relational model

- ERD
  - Entity
  - Attribute
  - Unique key
  - Relationship

- Relational model
  - Relational Table
  - Attribute (column)
  - Candidate key, primary key
  - Foreign key

From ERD to relational model

- Example:

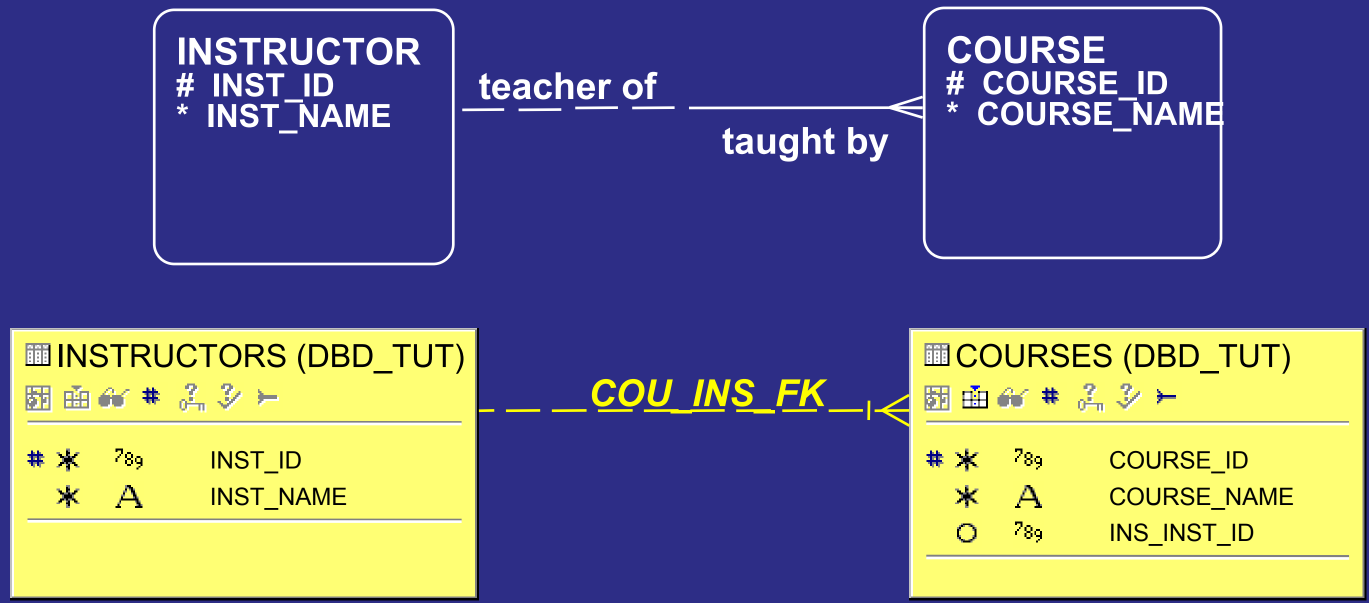


From ERD to relational model

- Binary 1:N relationships
  - Introduce a foreign key in the table on the “many” side
- Binary 1:1 relationships
  - Introduce a foreign key in the table on the optional side

1:N Relationships

- take the primary key of the table on the "1" side and insert it as a foreign key into the table on the "N" side



Normalisation

- Objective
  - validate and improve a logical design, satisfying constraints and avoiding duplication of data
- Is a process of decomposing relations with anomalies to produce smaller well-structured tables

Normal Forms

- Normalisation process
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce/Codd Normal Form (BCNF)
  - Higher Normal Forms (4NF, …)
- Usually the 3NF is appropriate for real-world applications

Functional dependency

- Is a relationship between attributes
- Notation: $A \rightarrow B$
  - If attribute B is functionally dependent on attribute A, then for every instance of A you can determine the value of B
  - Ex: empno $\rightarrow$ name, empno $\rightarrow$ job

Normalisation: 1NF

- All table attributes values must be atomic (multi-values not allowed)

- By definition a relational table is in 1NF

Normalisation: 1NF

Student(SID, SNAME, CID, CNAME, GRADE)

SID	SNAME	CID	CNAME	GRADE
224	Waters	M120	Database Management	A
		M122	Software Engineering	B
		M126	OO Programming	B
421	Smith	M120	Database Management	B
		M122	Software Engineering	A
		M125	Distributed Systems	B

Violation of the 1NF!

Normalisation: 1NF

SID	SNAME	CID	CNAME	GRADE
224	Waters	M120	Database Management	A
224	Waters	M122	Software Engineering	B
224	Waters	M126	OO Programming	B
421	Smith	M120	Database Management	B
421	Smith	M122	Software Engineering	A
421	Smith	M125	Distributed Systems	B

Normalisation:2NF

- 1NF
- Every non-key attribute is fully functionally dependent on the primary key (no partial dependencies)
  - No attribute is dependent on only part of the primary key, they must be dependent on the entire primary key

Normalisation:2NF

- Partial dependency

SID	SNAME	CID	CNAME	GRADE
224	Waters	M120	Database Management	A
224	Waters	M122	Software Engineering	B
224	Waters	M126	OO Programming	B
421	Smith	M120	Database Management	B
421	Smith	M122	Software Engineering	A
421	Smith	M125	Distributed Systems	B

Normalisation:2NF

- Decompose to 2NF:
  - For each attribute in the primary key that is involved in partial dependency, create a new table
  - All attributes that are partially dependent on that attribute should be moved to the new table

Student(SID, CID, ~~SNAME, CNAME,~~ GRADE)

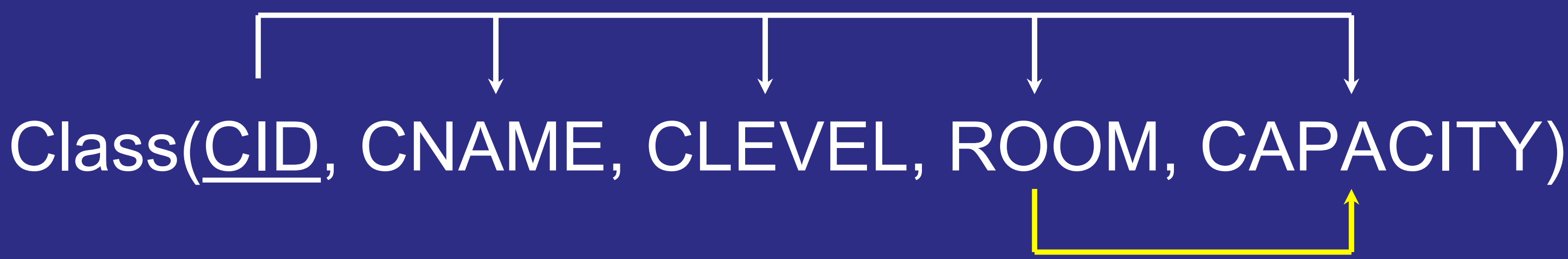
Student(SID, SNAME) Class(CID, CNAME)

Normalisation: 3NF

- 2NF
- No transitive dependency for non-key attributes
- Transitive dependency:
  - When an non-key attribute is dependent on another non-key attribute

Normalisation: 3NF

- Example:



Violation of the 3NF!

Normalisation:3NF

- Decompose into 3NF
  - For each non-key attribute that is transitive dependent on a non-key attribute, create a table

Class(CID, CNAME, CLEVEL, ROOM, CAPACITY)

→ Class(CID, CNAME, CLEVEL, ROOMID)

→ Room(ROOMID, CAPACITY)

Data access with SQL

- Structured Query Language
- Language used to create, manipulate and maintain a relational database
- Official ANSI Standard language for RDBMS access

SQL Terminology

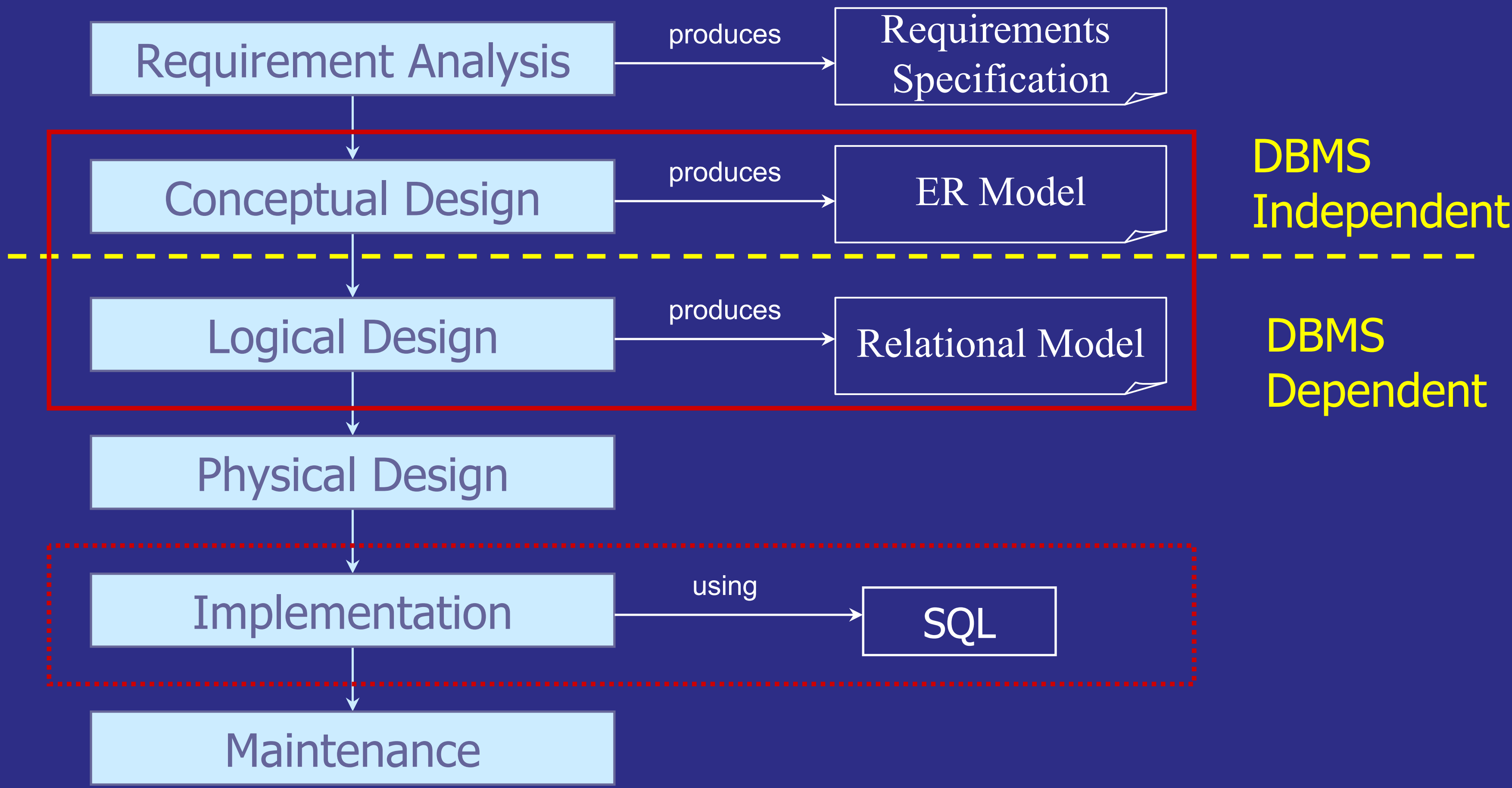
- **Data Definition Language (DDL)**
  - Define the database
  - CREATE, ALTER, or DROP a TABLE
- **Data Manipulation Language (DML)**
  - Manipulate the data in an existing database
  - SELECT, INSERT, UPDATE, or DELETE
- **Data Control Language (DCL)**
  - Control user access to an existing database
  - GRANT or REVOKE user privileges

Database schema

- Every user has a schema (account)
- References to an object that does not belong to the same schema must be prefixed it with the schema name
- Example:
  - SELECT * FROM emp;
  - SELECT * FROM marta.dept;

Note: user marta must grant privileges on the table dept to user demo.

What did we cover?



Documentation

- Oracle Design

  Dave Ensor, Ian Stevenson

  O'Reilly & Associates; ISBN: 1565922689; (April 1997)
- Oracle SQL: The essential reference

  David Kreines, Ken Jacobs

  O'Reilly & Associates; ISBN: 1565926978; (October 2000)
- Oracle online resources: http://otn.oracle.com

The end…

- Thank you for your attention!

- Consultancy on database design

  Physics-database.support@cern.ch

Questions