Engineering Databases

Lecture 3 – SQL and The Relational Model

November 02, 2022

M. Saeed Mafipour & Mansour Mehranfar

SQL lecture 2

- The **Data Definition Language** (DDL) is used to define and modify schema
- **Data Manipulation Language** (DML) is used to add, remove, and query data
- SQL a **declarative language**, you define what you want and not how to compute it

- DDL:
  - create table <tableName> (<c1Name> <c1Type> <c1constrains> [, …]);
  - drop <tablerName1> [, …];
  - rename <tableName> to <newName>
  - alter table <tableName> <specifications: add, modify, change and drop>;
- DML (partial):
  - insert into <tableName> (c1Name [, …]) values (c1Value [, …])
  - select <columns or *> from <data source, e.g. tableName> where <Conditions>

Data manipulation language

- Comprises methods to change and query data
- INSERT INTO statement
- SELECT statement
- UPDATE statement
- DELETE statement

Change records with SQL – UPDATE

- Changes values of an existing entity in a table
- The syntax is as follows
  - UPDATE <table name>

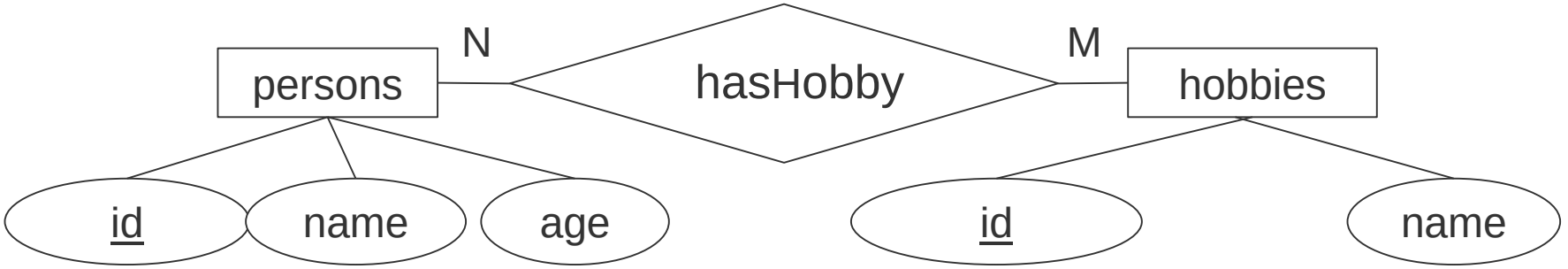
    SET <col name>=<col value> [,<col name>=<col value>,…]

    [WHERE <condition>]

- If no WHERE condition is given, all records are updated

- Example:
  - UPDATE instructors SET department='Cms' WHERE name='Bob'

SQL Exercise 1



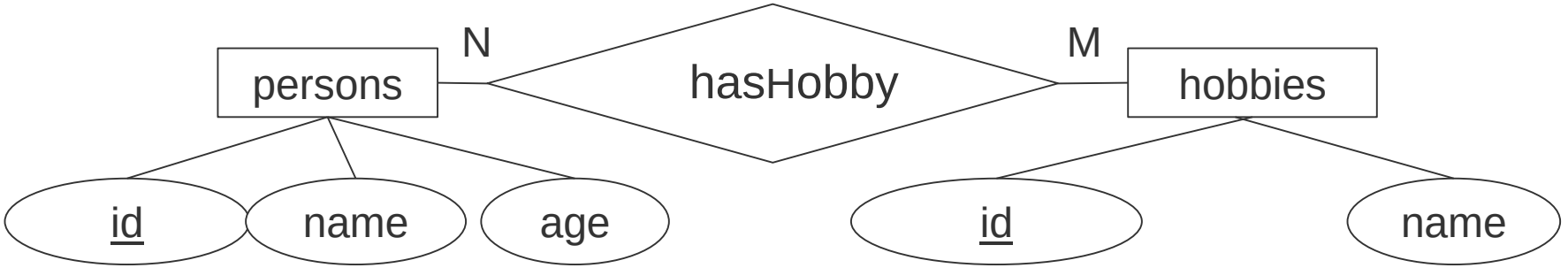
- If not already done:
  - Create tables (find the sql statements in moodle, lecture 2)
  - Insert two persons, two hobbies and three hasHobby data sets (sql - moodle)

- Update the age of all persons and set it to 40
- Update the hobby with id '1‘ and set the name to 'cycling'
- Update the person with name ’Peter‘ and set the name to ’Daniel‘ and the age to 31

Data removal with SQL – DELETE FROM

- Removes one or more records from a table
- The syntax is as follows
  - DELETE FROM <table_name> [WHERE …]
  - WHERE <columnName>=<value> [and | or <columnName>=<value>]
- If WHERE is not specified, all records are deleted
- Example
  - DELETE FROM instructors WHERE name='Bob' and department='Cms'
  - DELETE FROM instructors

SQL Exercise 2



- Delete all records in hasHobby
- Delete all records in hobbies that have the id '1'
- Delete all records in table persons that have the name ‘Daniel’ and age 31
- Hint: Combine statements following where by AND or OR

Additional: data insertion with SQL – INSERT INTO

- You can insert multiple rows in a single INSERT INTO statement
- Both insert methods (without and with column names) can be used
- The syntax is as follows

  INSERT INTO <table_name> VALUES

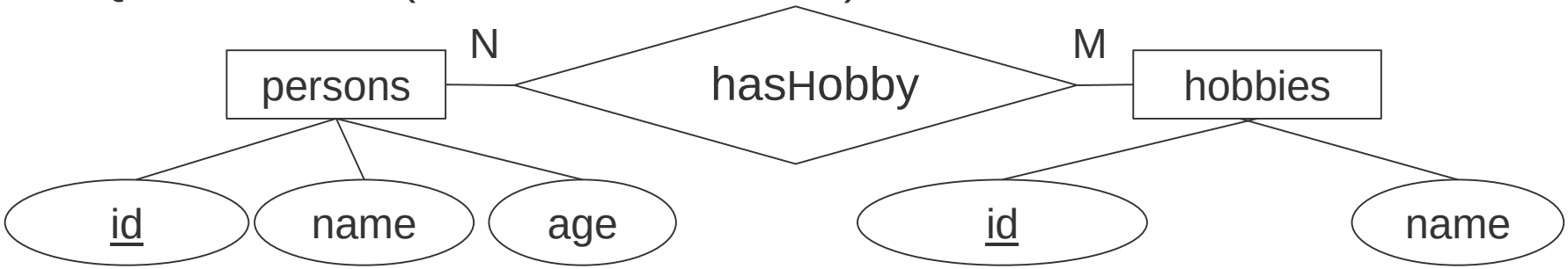
  (<value1>,<value2>,<value3>,...), (<value4>,<value5>,<value6>,...) [, …];

  INSERT INTO <table_name> (<column1>,<column2>,<column3>,...) VALUES

  (<value1>,<value2>,<value3>,...), (<value4>,<value5>,<value6>,...) [, …];

- Example:
  insert into instructors (department, name) values ('Cie', 'John'), ('Cms', 'Kate')
  insert into instructors ('Tom', 'Cie'), ('Jane', 'Cms')

SQL Exercise 3 (Students, 5 minutes!)

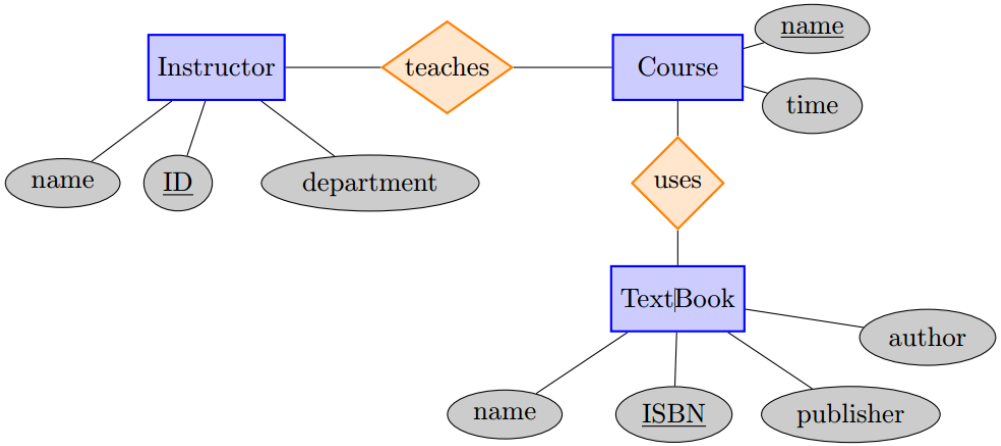


- **Add two new persons in a single insert into statement naming the columns**
  Hint: Make sure you use new ids that do not exist
- **Modify your hoobies table so that id has auto_increment**
  Hint: modify will remember old integrity constrains, but you have to add the type
- **Add two new hoobies in a single insert into statement without naming the columns**
  Hint auto_increment:
  Skip the column if you use the insert into with columns naming
  If you do not name the columns, use default or null as value.
- **Drop table persons, hasHobby, and hobbies in a single statement!**

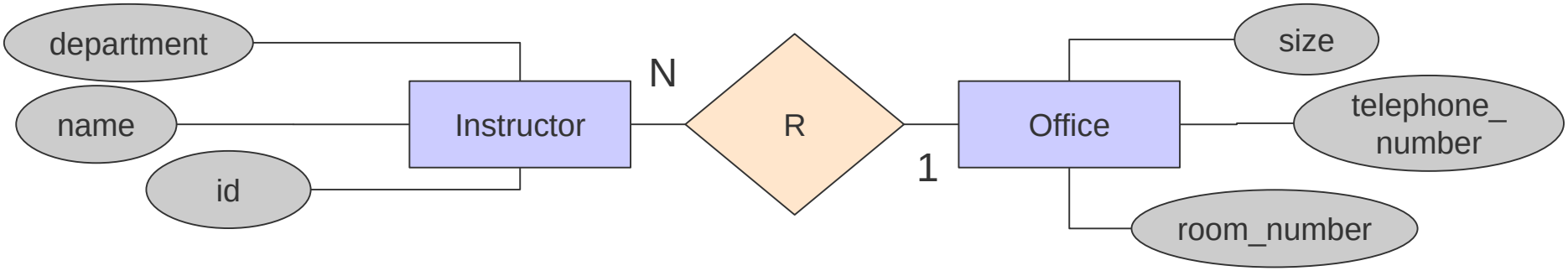
Mapping in ER-Schema onto a relational schema

Simple Rule

- each entity => relation (a table)
- each relationship => relation (a table)

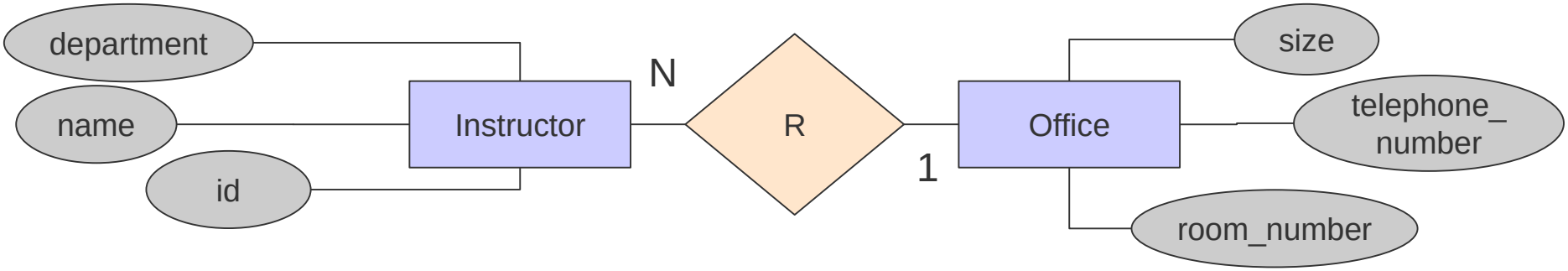


Refining the relational schema



- **Initial design: 3 relations**
  Instructor: {id, name, department}
  Office: {roomNumber, telephone_number, size}
  Instructor_has_Office: {id, room_number}
- Eliminate some of the relations introduced for relationships
  Only for: 1:1, 1:N, N:1 relationships → Merge relations with identical key
- **Resulting design: 2 relations**
  Instructor: {id, name, department, room_number}
  Office: {room_number, telephone_number, size}

Refining the relational schema



1
Design

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS

staff_number	room_number
1	5
2	10
3	25

room_number	telephone_number	size
5	1211	25.2
10	1213	20.1
25	1313	30

The Relational Model

- Introduced by Codd in 1970
  Classic paper:" A relational model for large shared databanks"
- Uses mathematical concept of relation
- Set theory and first-order predicate logic
- Commercial software
  - IBM DB2, Informix
  - Oracle
  - Microsoft SQL server, MS Access
- Open source
  - MySQL
  - PostgreSQL
  - SQLite

Relational Algebra

- Algebra = Types + Operations
- An algebra must be closed. Results of operations are of defined type.
- E.g. Boolean Algebra:
  - Boolean Type: TRUE/FALSE
  - Operators: AND, OR, NOT (=> XOR, NAND, NOR, XNOR)
- E.g. relational Algebra
  - Type = relation
  - Operations: selection, projection, join,…
- What for?
  - To retrieve data from the database
  - Meaningful combinations of different relations
  - Formal basis of SQL

Relational Algebra

- Formal definitions of

  Projection Π

  Union ∪

  Selection σ

  (More in Lecture 4)

Tables for the relational algebra exercises

- Create the following tables and data sets for the upcoming examples:

  - **create table instructors** (id integer primary key auto_increment, name varchar(40), department varchar(50))
  - **insert into instructors** (name, department)
    values ('Tom','CMS'),('Alice','CiE'), ('Bob','CMS')
  - **create table professors** (id integer primary key auto_increment, name varchar(40), department varchar(50))
  - **insert into professors** values (null,'Rank','CiE'), (null,'Borrman','CMS');

Projection

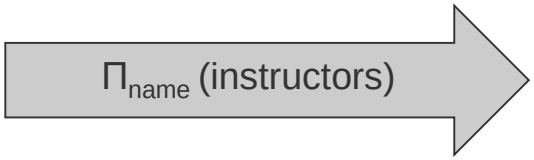
In general: $\Pi_{columns}$ (Relation)

- Extracts columns (attributes) from a table
- Result contains (all) data from specified columns

- Example: Π_{name} (instructors)

```
SELECT name FROM instructors
```

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS



name
Tom
Bob
Alice

Projection

Some rules of the $\Pi_{columns}$ (Relation) operator

- Π_a(Relation) = Π_a(Π_a(Relation))
- Π_a(Π_b(Relation)) = Π_a(Relation) if $a \subseteq b$

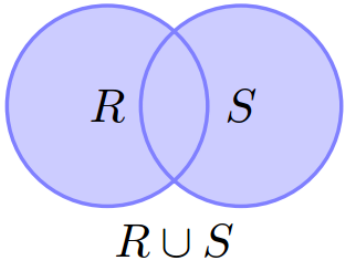
id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS

- Example of a nested statement:
  SELECT department from (select department from instructors) as instructorAlias

  The result of (…) is named instructorAlias
- Exercise (Students → 7 mins):
  $\Pi_{department}$(instructors),
  Π_{name} (Π_{name} (instructors)),
  Π_{name} ($\Pi_{name, department}$ (instructors)),
  $\Pi_{name, department}$ ($\Pi_{department}$ (instructors)) (error! why?)
  $\Pi_{name, department}$ (Π_{id} (instructors)) (error! why?)

Union

- In general: $R_i \cup R_j$
- Two relations with identical schema be merged to one
- Example:



instructors ∪ professors

SELECT * FROM professors UNION SELECT * FROM instructors

$\Pi_{\text{name, department}}$ (instructors) ∪ $\Pi_{\text{name, department}}$ (professors)

instructors relation

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS

professors relation

id	name	department
10	Borrmann	CMS
30	Rank	CiE

Union

id	name	department
10	Borrmann	CMS
30	Rank	CiE
1	Tom	CMS
3	Bob	CMS
2	Alice	CiE

- Exercise: Create a union of the names of the tables instructors and professors

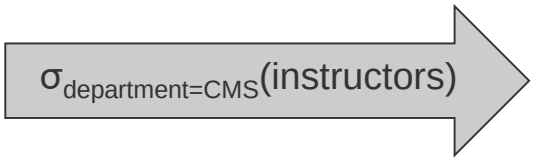
Selection

In general: $\sigma_{statement}(Relation)$

- Selection predicate (statement) is a formula containing
  - Attribute names
  - Comparison operators: =, <, <= , >, >=, !=
  - Logical operators
- Result contains all tuples t ∈ Relation for which formula is fulfilled
- Example: $\sigma_{department=CMS}(instructors)$

```
SELECT * FROM instructors WHERE department='CMS'
```

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS



id	name	department
1	Tom	CMS
3	Bob	CMS

Selection

Some rules of the $\sigma_{statement}(Relation)$ operator

- $\sigma_a(Relation) = \sigma_a(\sigma_a(Relation))$
- $\sigma_a(\sigma_b(Relation)) = \sigma_b(\sigma_a(Relation))$
- $\sigma_{a \text{ and } b}(Relation) = \sigma_a(\sigma_b(Relation))$
- $\sigma_{a \text{ or } b}(Relation) = \sigma_a(Relation) \cup \sigma_b(Relation)$

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS

- Example for nested statements:
  SELECT * FROM (SELECT * FROM instructors WHERE department= 'CMS')
  AS instructorsAlias WHERE department='CMS'
- Exercise:
  1. $\sigma_{department=CMS}(instructors)$, 2. $\sigma_{department=CMS}(\sigma_{department=CMS}(instructors))$
  3. $\sigma_{department=CMS}(\sigma_{name=Tom}(instructors))$, 4. $\sigma_{name=Tom}(\sigma_{department=CMS}(instructors))$,
  5. $\sigma_{department=CMS \text{ and } name!=Tom}(instructors)$, 6. $\sigma_{department=CMS}(instructors) \cup \sigma_{name!=Tom}(instructors)$, and 7. $\sigma_{department=CMS \text{ or } name=Tom}(instructors)$

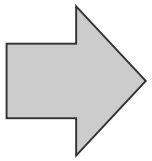
Combination of Selection and Projection

In general: $\Pi_{columns}(\sigma_{statement}(Relation))$ or $\sigma_{statement}(\Pi_{columns}(Relation))$

- Extracts columns (attributes) from a table for which the statement is fulfilled
- $\sigma_b(\Pi_a(Relation))$ will fail if $b \subseteq a$ is not given
- Example:

$\Pi_{name}(\sigma_{department=CMS}(instructors))$

id	name	department
1	Tom	CMS
2	Alice	CiE
3	Bob	CMS



name
Tom
Bob

$\sigma_{department=CMS}(instructors)$

id	name	department
1	Tom	CMS
3	Bob	CMS

$\Pi_{name}(instructors)$

name
Tom
Alice
Bob

$\sigma_{department=CMS}(\Pi_{name}(instructors))$

$\sigma_{department=CMS}(\Pi_{name,department}(instructors))$

name	department
Tom	CMS
Bob	CMS

Combination of Selection and Projection

- Homework:

  Execute the SQL statements

  $\Pi_{name}(\sigma_{department=CMS}(instructors))$

  - Select name from instructors where department $_{= CMS}$
  - Select name from (Select * from instructors where department $_{= CMS}$) as alias1

    $\sigma_{department=CMS}(\Pi_{name}(instructors))$ (error!)

    $\sigma_{id=1}(\sigma_{department=CMS}(\Pi_{id,department}(instructors)))$ – Result 1,CMS

    $\sigma_{id>2 \text{ and } department=CMS}(\Pi_{id,department}(instructors))$ – Result 3,CMS

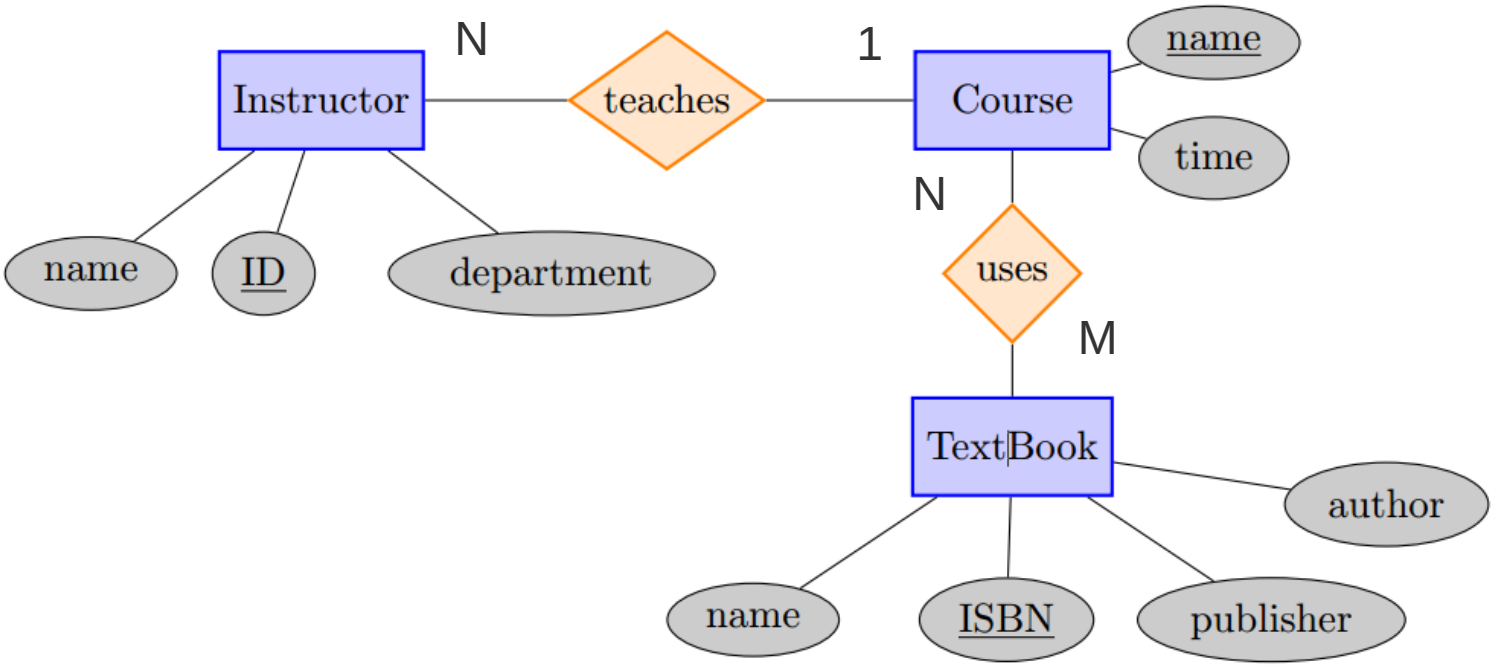
    $\sigma_{id>2}(\Pi_{department}(\Pi_{id,department}(instructors)))$

select id from (Select id, department from instructors where $_{id!=3 \text{ or } department=CMS}$)as alias1

select id from (Select $_{* \text{ from}}$ (Select id, department from instructors) as alias1 where $_{id!=3 \text{ or } department=CMS}$) as alias2

Mapping in ER-Schema onto a relational schema

- Homework:
  Define relations (tables) based on the simple and advanced mapping design



End of Lecture

Thank you for your attention