Relational Databases: Reminders

Responsible:
Dr. Tegawendé F. BISSYANDE
tegawende.bissyande@uni.lu

Course Author:
Eduardo Cunha de Almeida
eduardo@inf.ufpr.br

Teacher Assistant:
Médéric Hurier
mederic.hurier@uni.lu

Course Features

- Sep. 20th - **Introduction to Big Data**

Part 1. Databases and Query Models for Big Data

- Sep. 27th - **Relational Databases: Reminders**
- Oct. 4th - **Relational Databases: Internals**
- Oct. 11th - **NoSQL Databases**
- Oct. 18th - **MapReduce Model**
- Oct. 25th - **Hadoop and Spark**
- Nov. 8th - **Datalog Model**

Part 2. Data Analysis and Machine Learning

- Nov. 15th - **Statistics and Probabilities**
- Nov. 22th - **Communication and Visualization**
- Nov. 29th - **Features Engineering and Supervised Learning**
- Dec. 5st - **Unsupervised and Reinforcement Learning**
- Dec. 12th - **Homework Time**

Section Features

- Definitions
- Conceptual Design
- Relational Model
- From Conceptual to Relational
- Structured Query Language (SQL)

Definitions

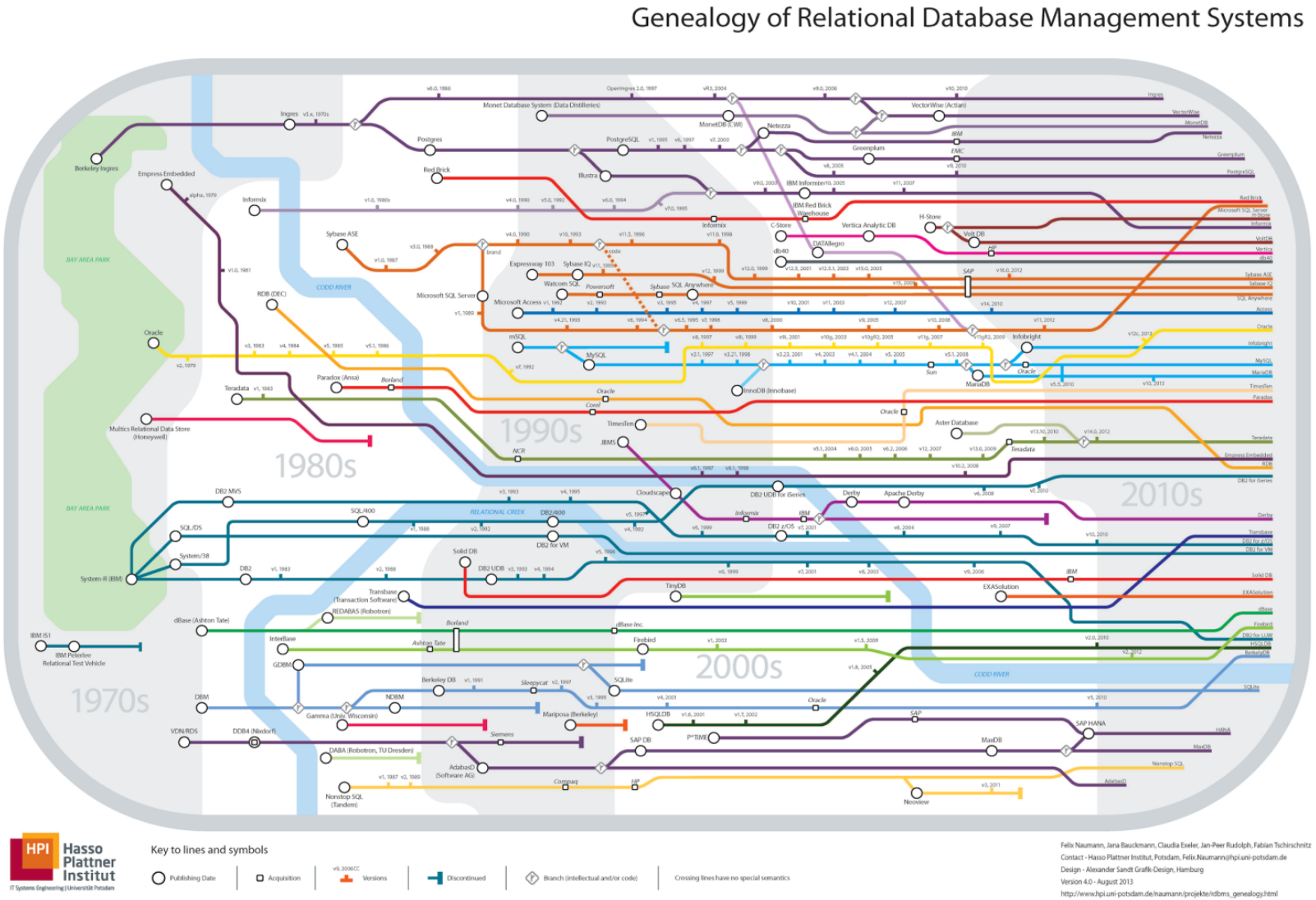
What is a database?

A database is an organized collection of data
[Navathe and Elmasri, 94]

- has coherence (no random sets)
- represents aspects of reality
- is built for specific projects

Database Genealogy

[Felix Naumann, 13]



Database VS File System

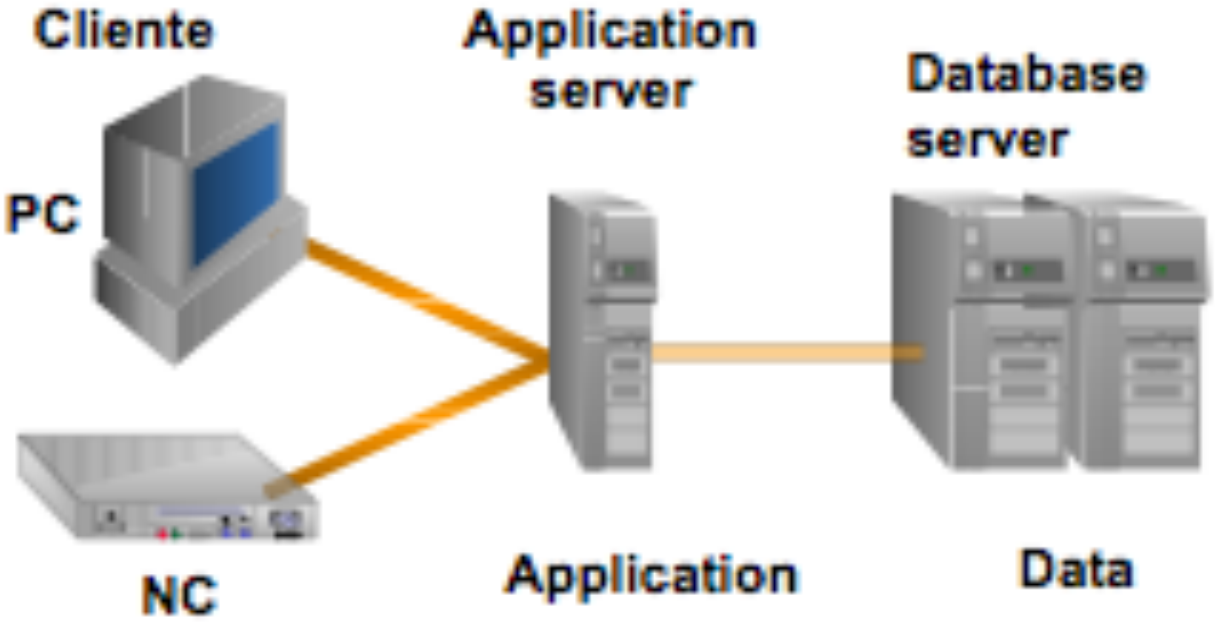
- **Self-descriptive nature:**
  - DB defines its data structures and constraints
- **Data abstraction:**
  - DB does not require programs to describe data
- **Multiple view:**
  - DB has different perspective to visualize data
- **Sharing:**
  - DB allows concurrent access

Database Actors

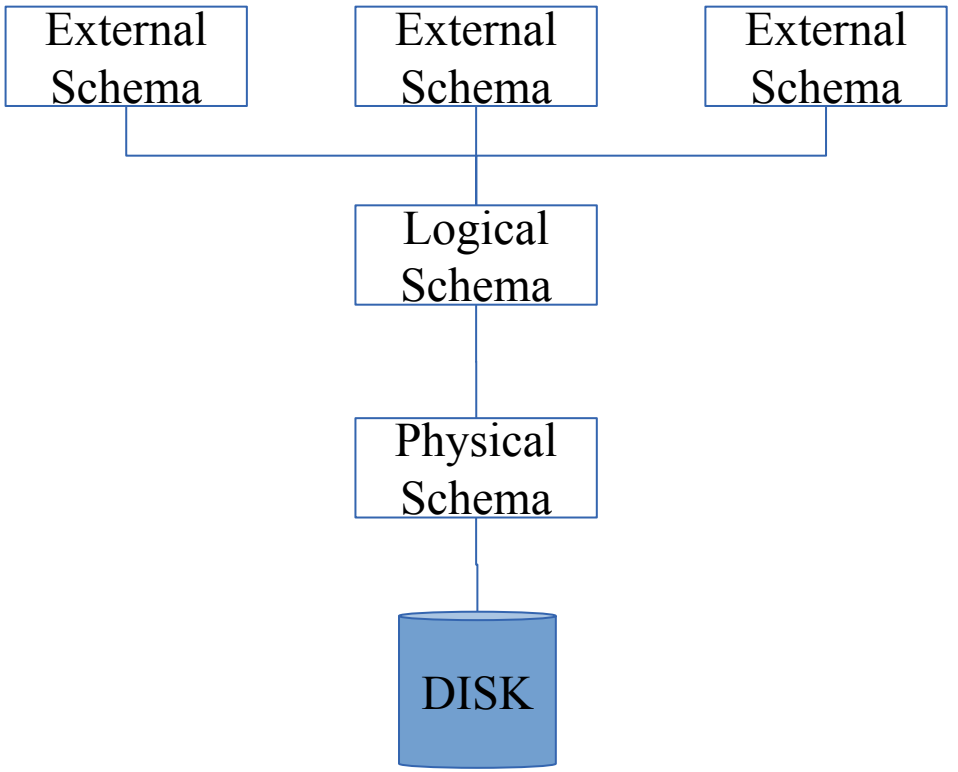
- **Users:**
  - Works on top of databases
- **Analyst:**
  - Determines the users' requisites
- **Designer:**
  - Designs the DB for specific projects
- **Database Administrator (DBA):**
  - Manages the database structures and resources

3-Tiers Architecture: Client/Server

[Valduriez, 92]

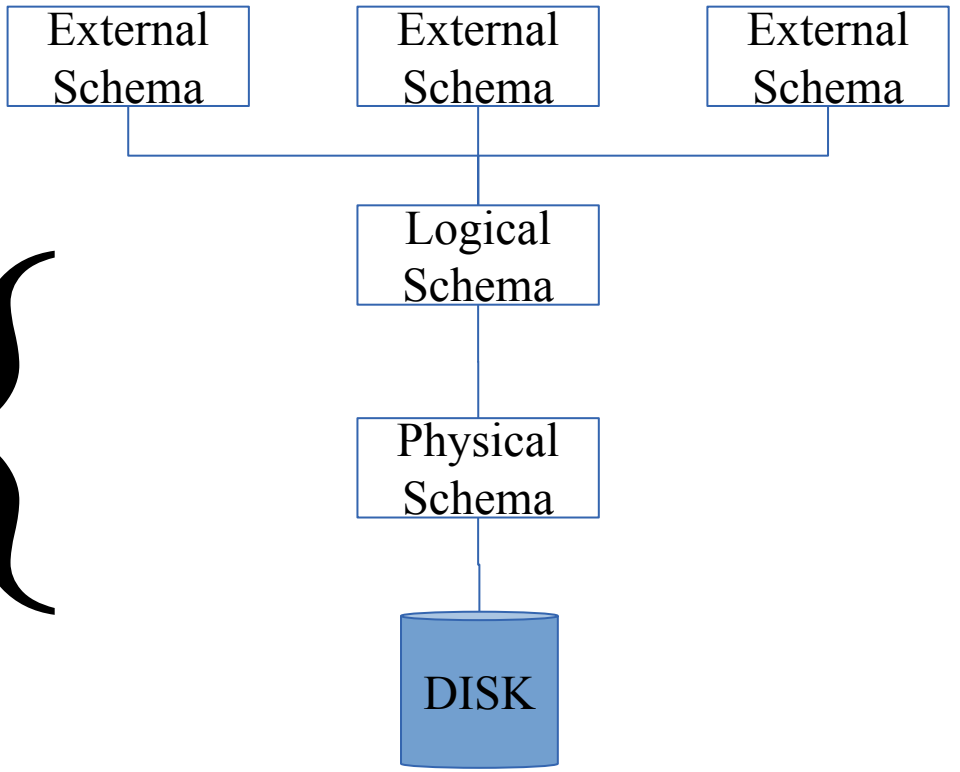


Database Abstraction



Today's lecture

- Conceptual Design
- Relational Model
- E-R Mapping



Conceptual Design

Why do we need database design?

Agree on structure the database before implementation

- Entities
- Relationships
- Constraints of the domain

Conceptual Design

Entity/Relationship Model (E/R)

Requirements => Design => Implementation

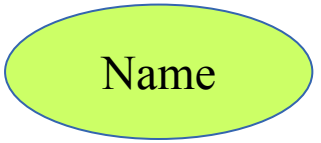
Different from UML conceived to support OO design !!!

E/R Diagrams

- Entities

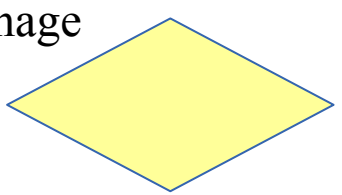
  - "Something" from the real word with independent existence

- Attributes



  - Properties of an entity

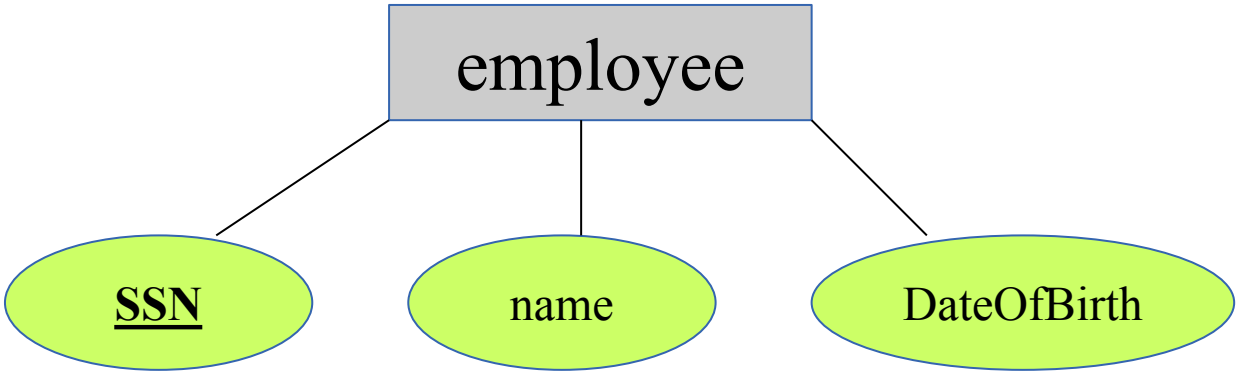
- Relationship



  - Association between entities

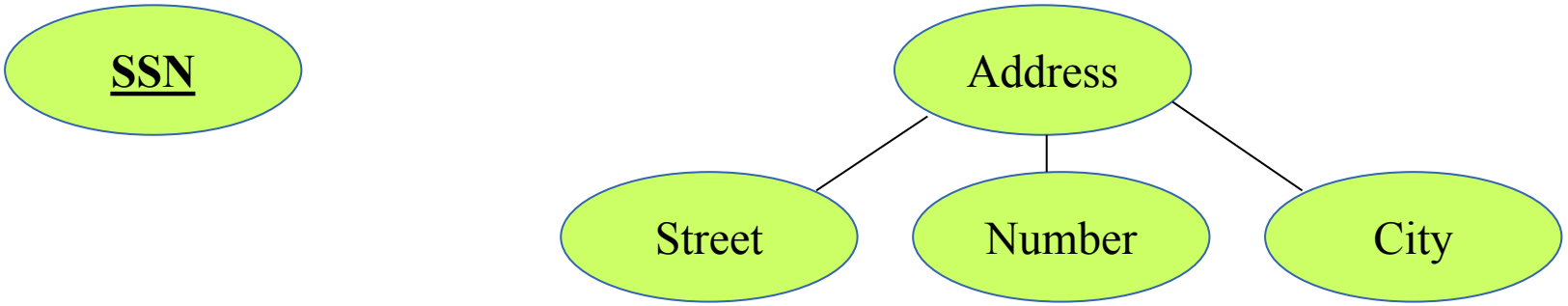
Entities, attributes and keys

Every entity has a minimal set of uniquely identifying attributes (i.e <u>key</u>)



Types of attributes

- Simple or Composite

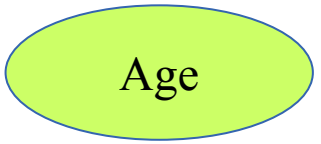


- Multivalued

  - e.g. Ph.D., M.S., B.S.

Types of attributes

- Derived



  - Requires some computation

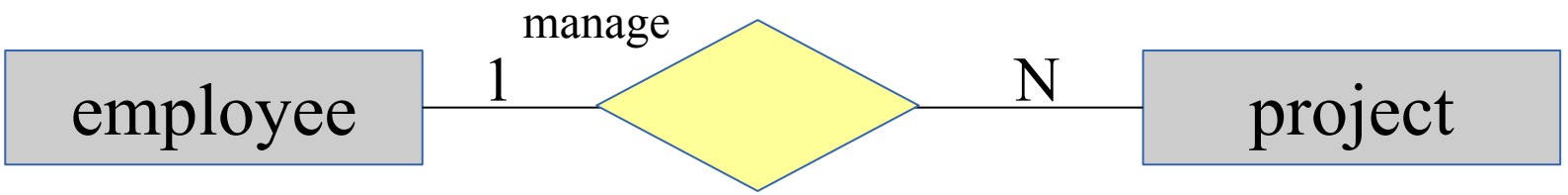
- Key

  - Identify uniquely an entity

Relationships

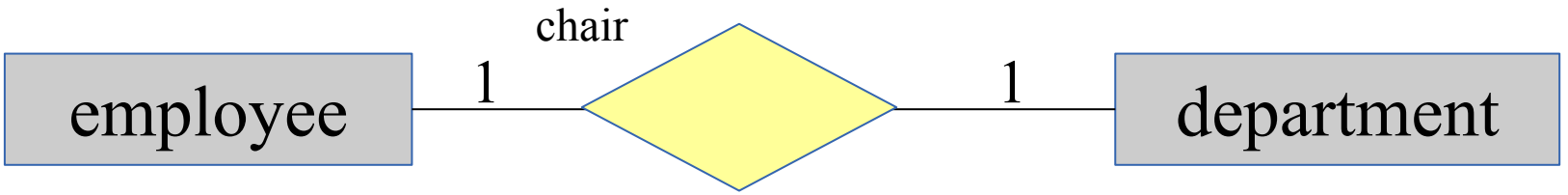
Connect entities together
(in general identified by a verb)

- 1:N relationship (the norm in the design)

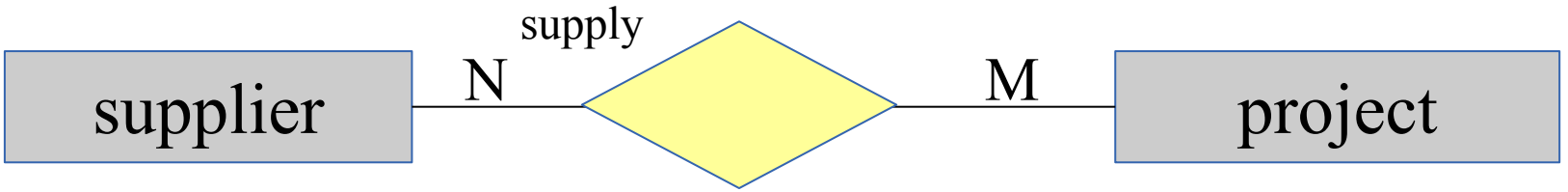


Relationship cardinality

- 1:1 relationship (rare, may belong to the same table)

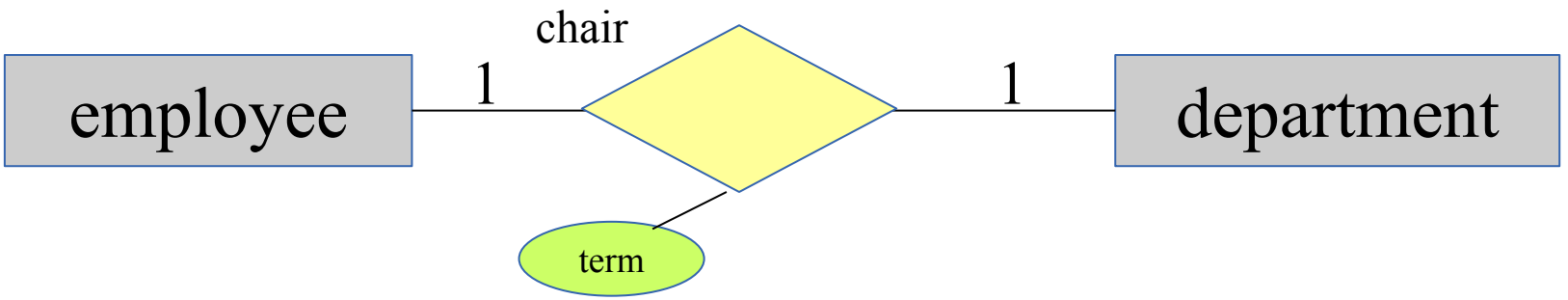


- N:M relationship (not so rare, but try to avoid)

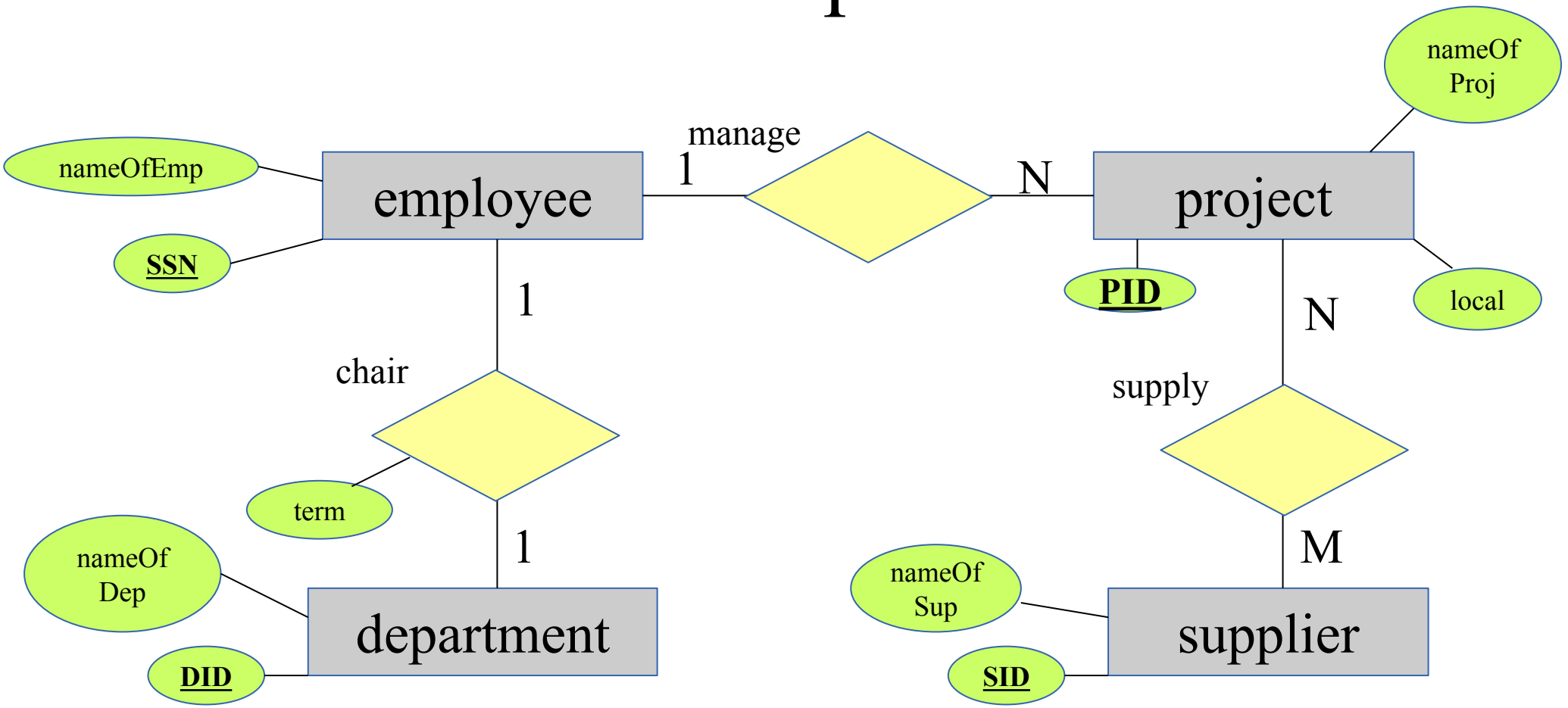


Relationship's Attributes

An attribute of a relationship only exists due to such association

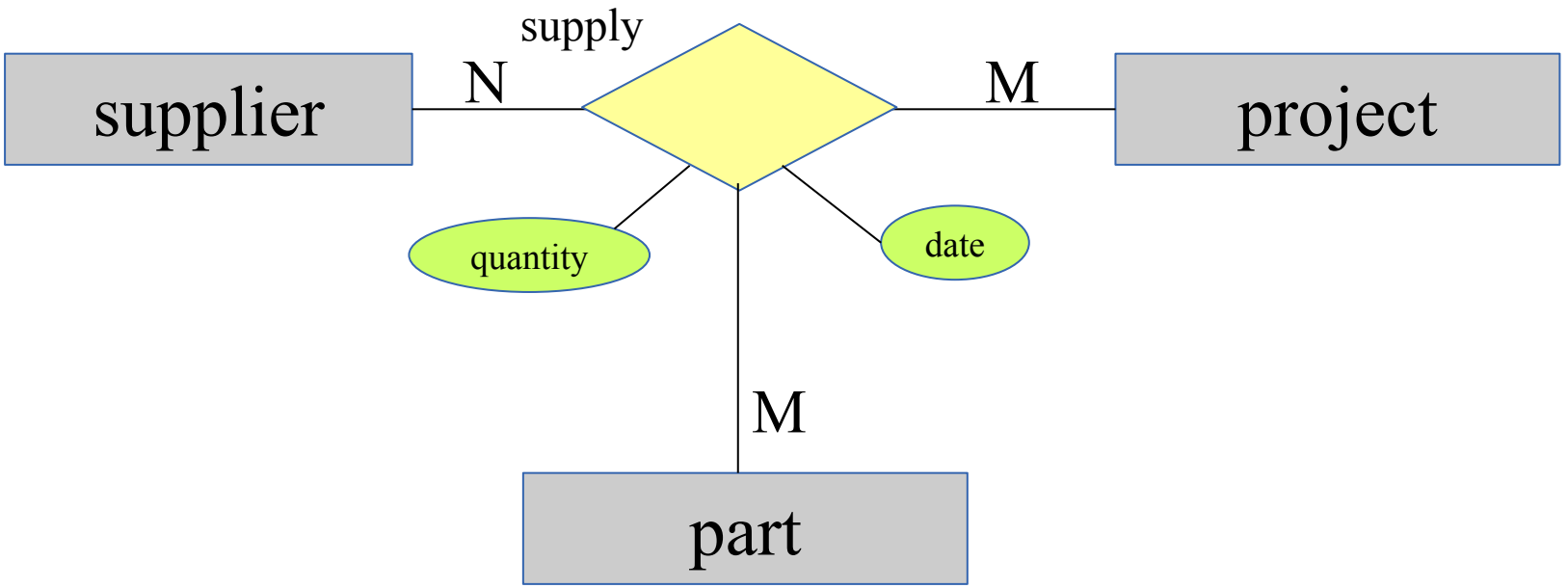


Example


nameOfEmp
employee
SSN
manage
1
N
project
nameOf
Proj
PID
local
1
N
chair
supply
term
1
M
nameOf
Dep
department
DID
nameOf
Sup
supplier
SID

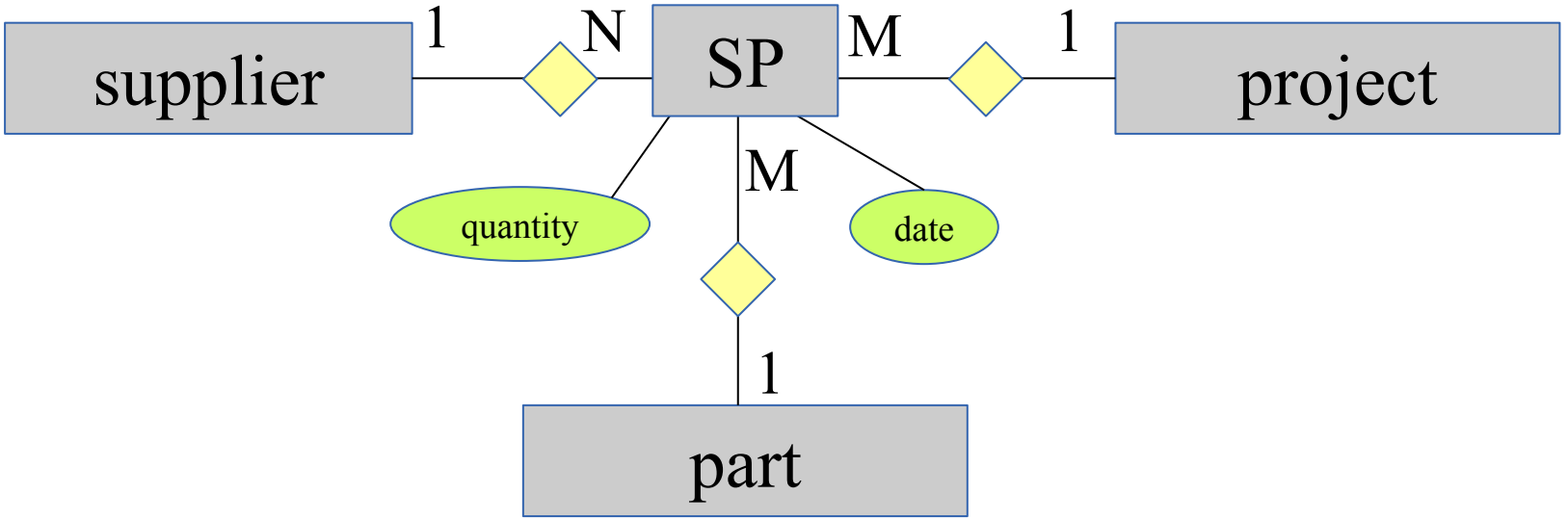
Multi-way Relationships

- Ternary relationship

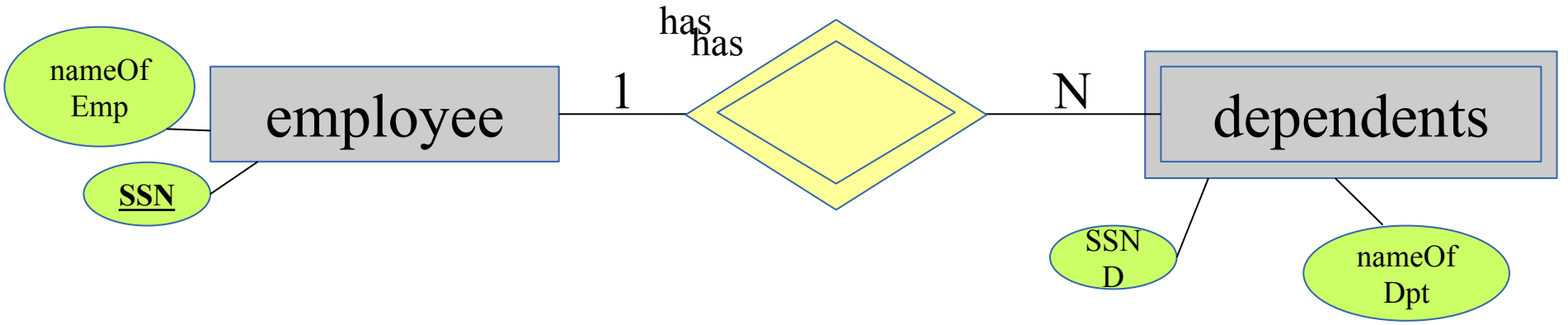


Multi-way Relationships

- Ternary to binary relationship



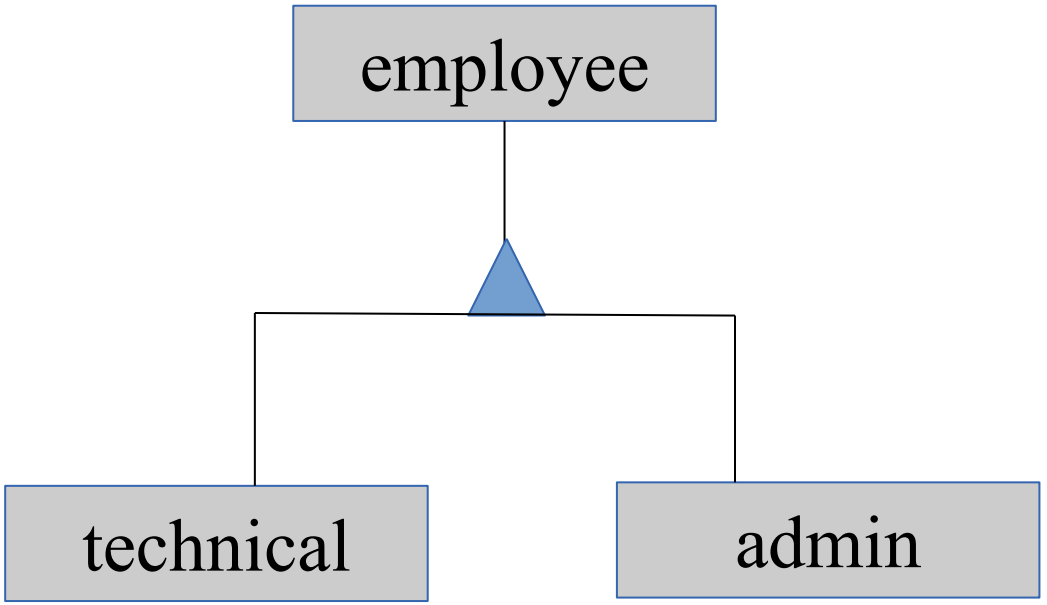
Weak Entities



- Cannot be identified by its attributes alone
- Requires a foreign key in conjunction with its attributes

Modeling Hierarchy

Data is naturally hierarchical (such as the world)

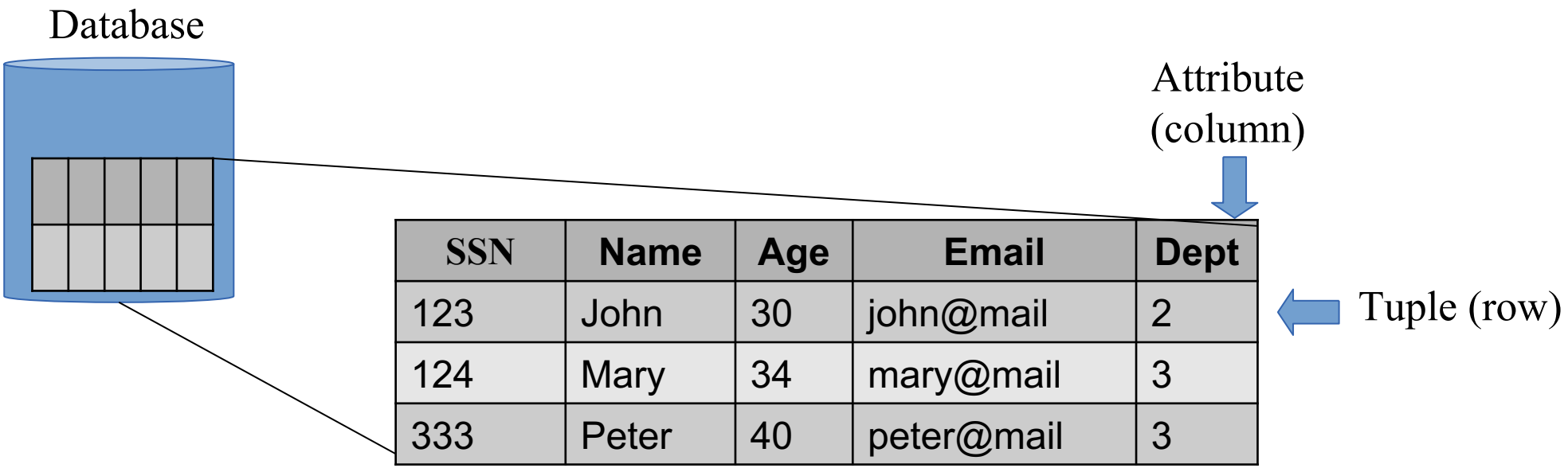


But mot all database systems implement inheritance !

Relational Model

Relational Model

- Created by Edgar Codd in 1969
- Based on mathematical relations
  - Set of tuples grouped into relations



SSN	Name	Age	Email	Dept
123	John	30	john@mail	2
124	Mary	34	mary@mail	3
333	Peter	40	peter@mail	3

Attributes (Columns)

- **Data type:**
  - integer, float, string, date/time, binary …

SSN	Name	Age	Email	Dept
123	John	30	john@mail	2
124	Mary	34	mary@mail	3
333	Peter	40	peter@mail	3

Domain

Constraints

- **Domain:** every element respects its domain
  - E.g., Dom(dept) = {1, 2, 3, 4, 5}

SSN	Name	Age	Email	Dept
123	John	30	john@mail	2
124	Mary	34	mary@mail	3
333	Peter	40	peter@mail	3

Domain

Constraints

- **Entity integrity:** No primary keys can be null

Primary Key

SSN	Name	Age	Email	Dept
123	John	30	john@mail	2
NULL	Mary	34	mary@mail	3
NULL	Peter	40	peter@mail	3

Tuples cannot be identified

Constraints

- **Referential Integrity:** enforces consistency between two relations

Table: Employee

SSN	eName
123	John
321	Mary
333	Peter

Table: Dependent

SSN	ID	dName
123	1	James
123	2	Diana
444	3	Robert

Broken link !

Examples of constraint violation

Table: Employee

SSN	Name	Age	Email	Dept
123	John	30	john@mail	2
124	Mary	34	mary@mail	3
333	Peter	40	peter@mail	3

- Insert(null, 'Gail', 32, gail@mail, 3) into employee
- Insert(123, 'Gail', 32, gail@mail, 3) into employee
- Insert(125, 'Gail', 32, gail@mail, 'A') into employee
- Insert(125, 'Gail', 32, gail@mail, 3) into employee

Database Normalization

The process of organizing a relational database to reduce data redundancy and improve data integrity

UNF, 1NF, 2NF, 3NF, 4NF, 5NF, 6NF

- A database is often described as "normalized" if it meets Third Normal Form (3NF)
- Most 3NF tables are free of insertion, update, and deletion anomalies

From Conceptual to Relational

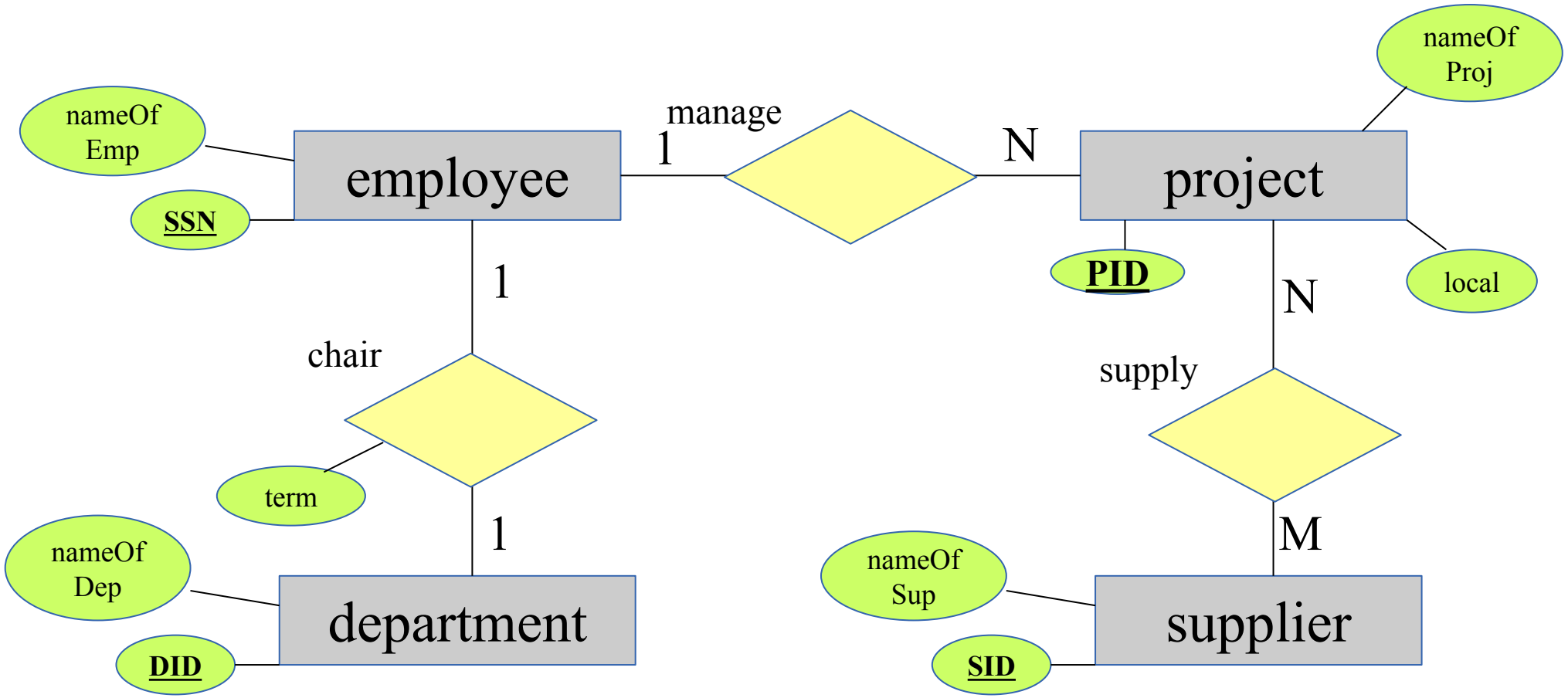
General Algorithm

1. Every entity becomes a relation with a key

1. Relationship 1:N sets a key to the N relation

1. Relationship N:M creates a new relation with the keys from both sides

Example


nameOf Emp
employee
SSN
manage
1
N
project
nameOf Proj
PID
local
1
chair
term
1
N
supply
M
nameOf Dep
department
DID
nameOf Sup
supplier
SID

Example

Table: project

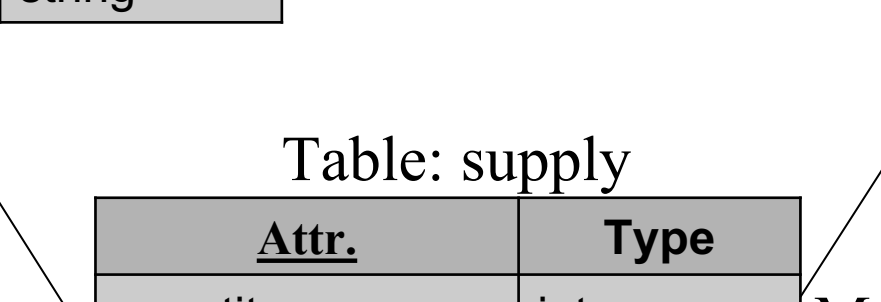
Attr.	Type
PID	integer
nameOfProject	string
local	string

Table: supplier

Attr.	Type
SID	integer
nameOfSup	string

Table: supply

Attr.	Type
quantity	integer
product	string
PID	integer
SID	integer



Structured Query Language (SQL)

SQL in a nutshell

- Programming language to manipulate data
- Declarative nature (what instead of how)
- Different versions:
  - 86, 89, 92, 99, 03, …
- Different aspects:
  - DDL (Definition), DML (Manipulation), DCL (Control), DTL (Transaction), DQL (Query)

SELECT

Select [attributes]
From [relation]
Where [condition]

For instance:

Select SSN, name
From Employee
Where age<40;

Select SSN, name
From Employee
Where age<40
and depto=2;

```
Select SSN, name
From Employee
Where age = 40
        and age = 50;
```

```
Select SSN, name
From Employee
Where age = 40
        or age = 50;
```

```
Select SSN, name
From Employee
Where age between 40 and 50;
```

- Consider the operators >, <, >=, <=

Joining Relations (Join operator)

[SQL92]

Select p.nameOfProj, s.quantity
From Project p, Supply s
Where p.pid=s.pid;

Select p.nameOfProj, s.quantity, r.nameOfSup
From Project p, Supply s, Supplier, r
Where p.pid=s.pid And s.sid=r.sid;

INSERT

Insert Into [relation]
Values (...)
Where [condition]

For instance:

Insert into
employee
Values (111, 'Jane', 45);

Insert into
employee(ssn, name, age)
Values (111, 'Jane', 45);

UPDATE

Update [relation]
Set (...)
Where [condition]

For instance:

Update employee
Set salary=200000;

Update employee
Set age=45
Where ssn=111;

DELETE

Delete From [relation]
Where [condition]

For instance:

Delete from employee
Where name= 'Jane';

Delete from employee
Where age between 0 and 18;

Thank You !
15 minutes break