Anthony Ventresque

Big Data Programming

COMP47470

SQL: Conceptual Design and Relational Model

School of Computer Science, UCD

Scoil na Ríomheolaíochta, UCD

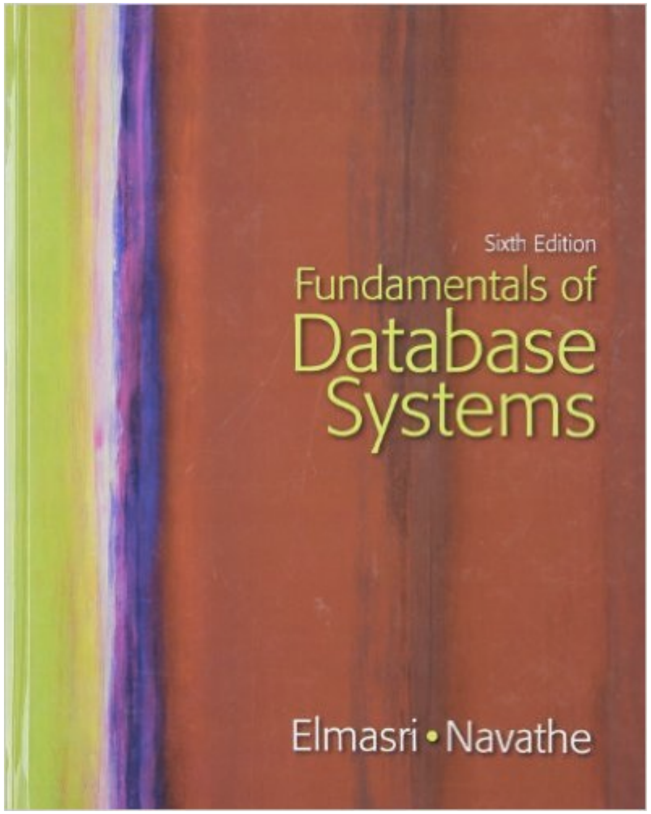
Last Week…

- Volume
- Velocity
- Variety
- (Veracity)
- Processing Big Data is skill- and labour-intensive

- CLI (bash) is an essential tool to prototype Big Data jobs, manage Big Data systems and execute Big Data queries

Outline

- Definitions
- Conceptual Design
- Relational Model



Take home message:

The Conceptual design (E/R model) is an agreement. The Relational model is a way to structure data in a DB (tables, constraints)

DEFINITIONS

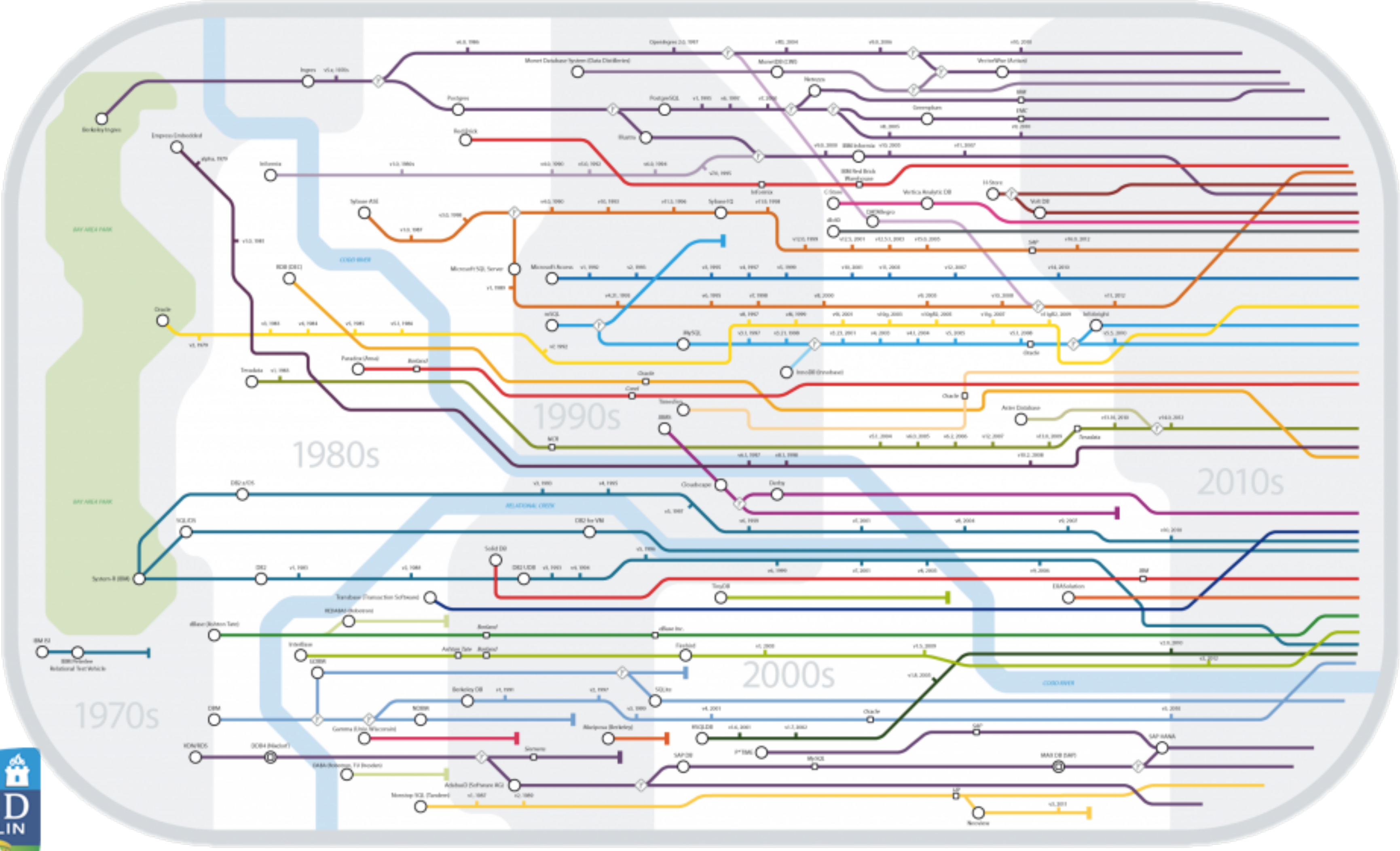
What is a Database?

A database is an organized collection of data

[Navathe and Elmasri, 94]

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

Database Genealogy



[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 views:
  - DB has different perspectives to visualise data
- Sharing:
  - DB allows concurrent access

Database Actors

- User:
  - 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-tiered Architecture: Client/Server

Presentation tier

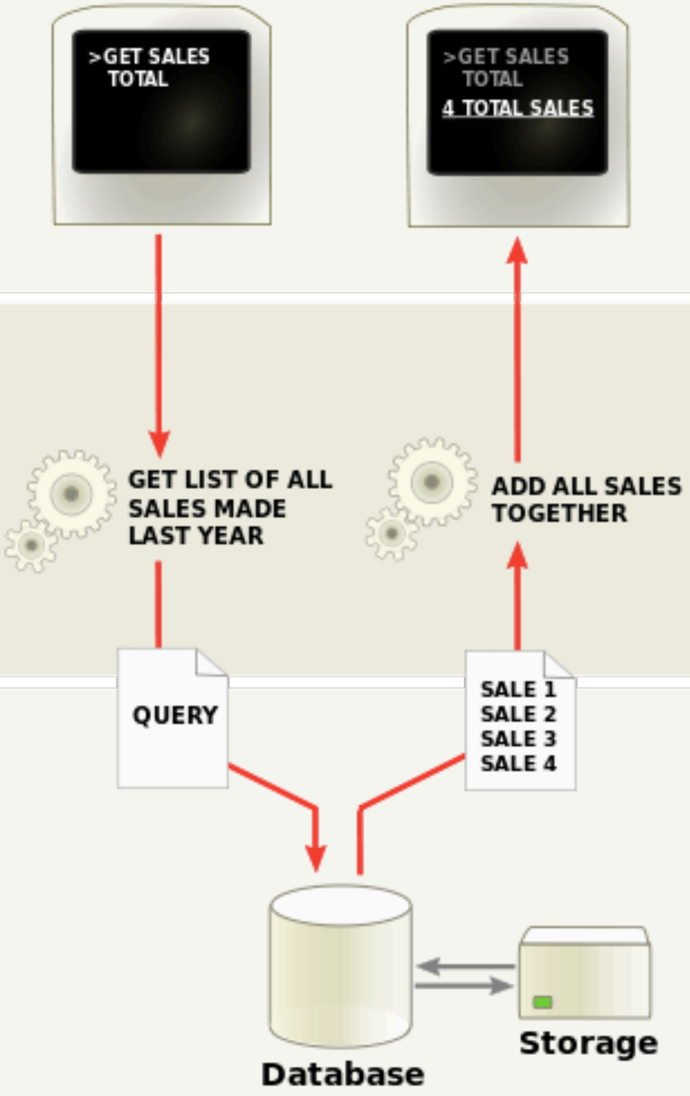
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

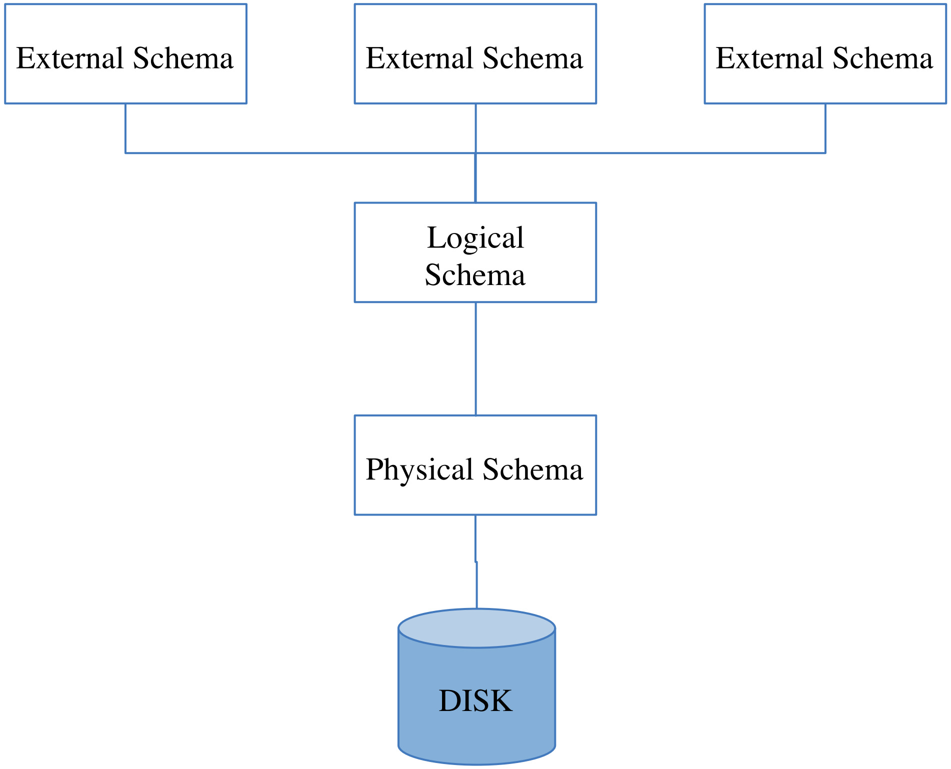
This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

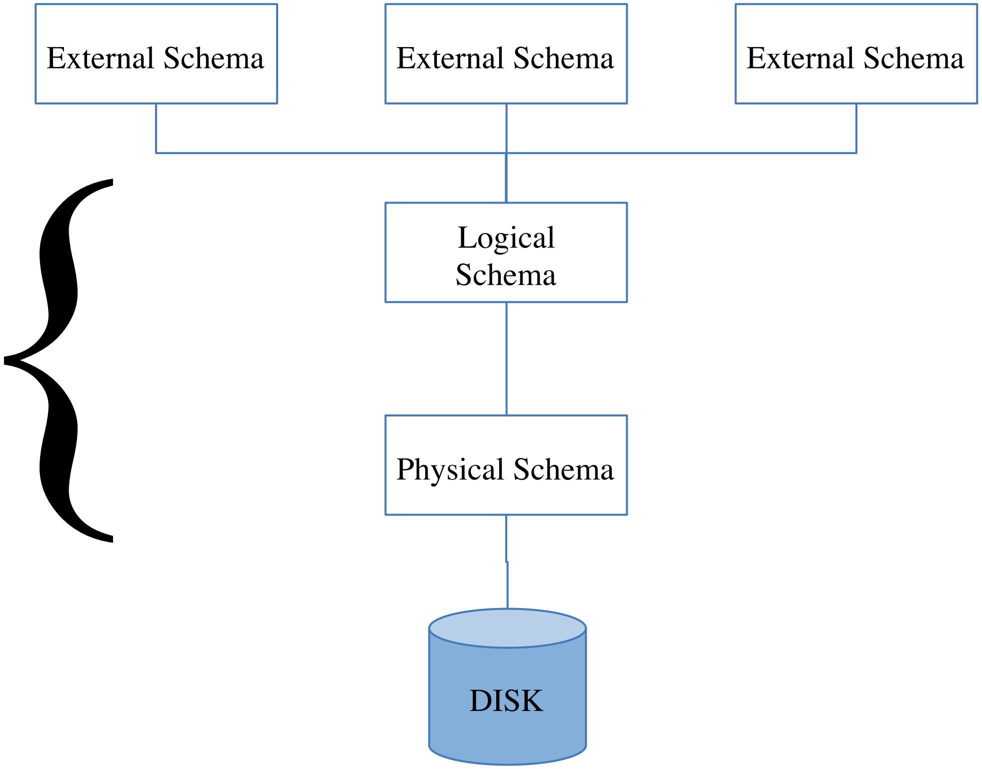


Database Abstraction



Lectures Today and Tomorrow

- Conceptual Design
- Relational Model
- (E-R Mapping)



CONCEPTUAL DESIGN

Why Do We Need Database Design?

Agree on the structure of the database before implementation

- Entities
- Relationships between entities
- Constraints of the domain

Conceptual Design

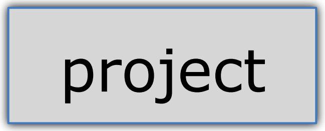
Entity/Relationship Model (E/R)

Requirements => Design => Implementation

Different from UML which aims at supporting OO design

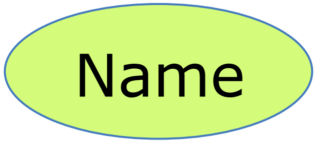
E/R Diagrams

Entities



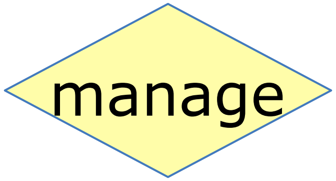
“Something” from the real word with independent existence

Attributes



Properties of an entity

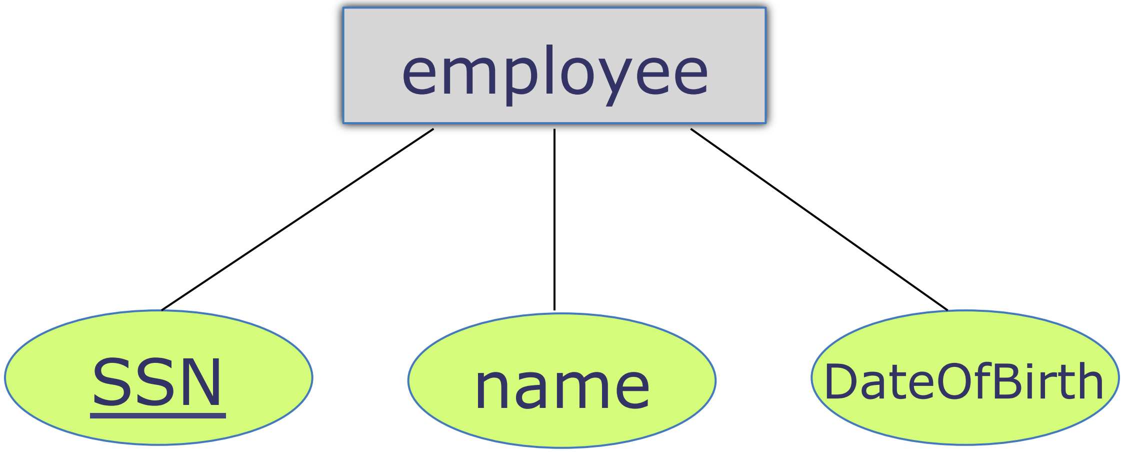
Relationship



Association between entities

Entities, Attributes and Keys

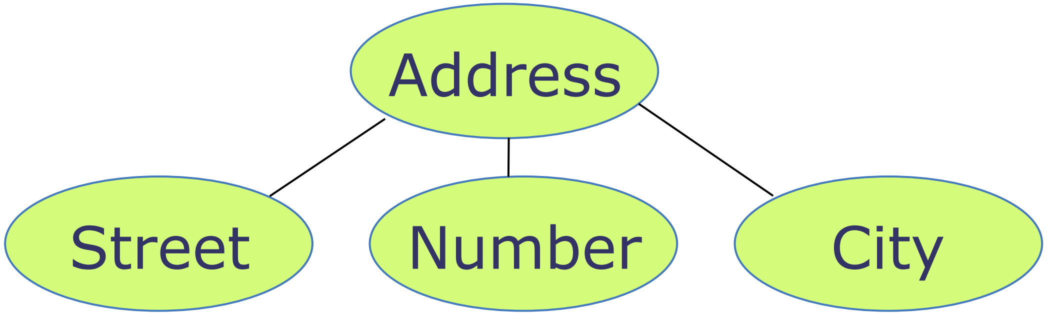
Every entity has a minimal set of uniquely identifying attributes (i.e., key)



Types of Attributes

- Simple or composite

SSN



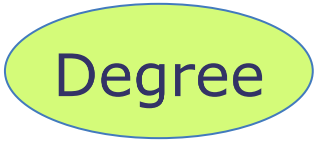
Address

Street

Number

City

- Multivalued

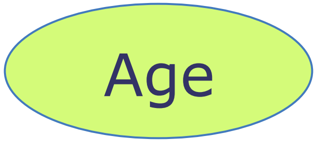


Degree

e.g., PhD, MSc, BSc

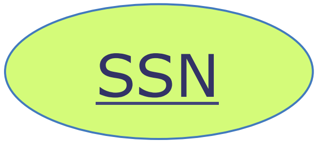
Types of Attributes

- Derived



Requires some computation

- Key

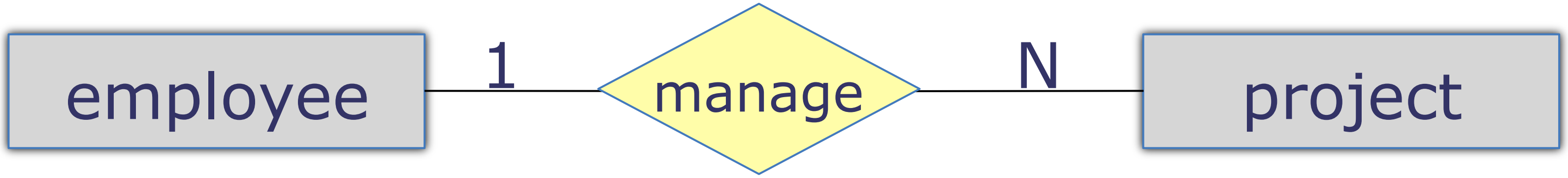


Identify uniquely an entity

Relationships

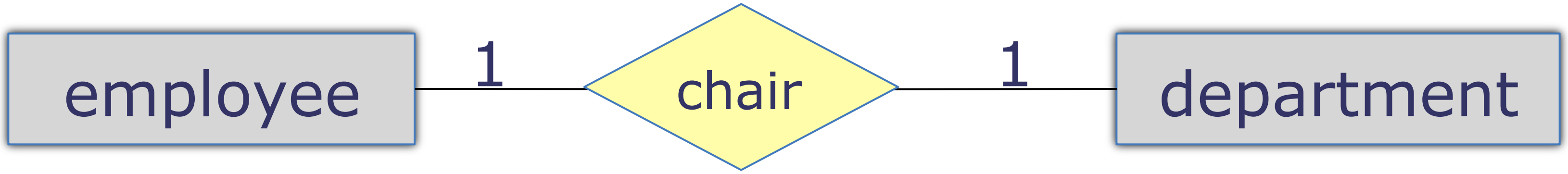
Connect entities together (in general identified by a verb)

- 1:N relationship (the norm)

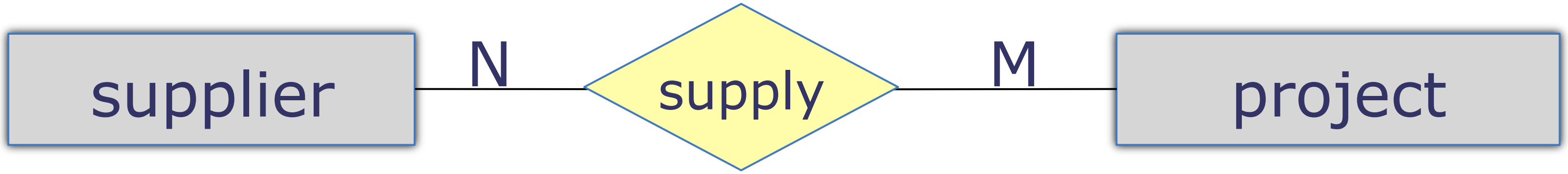


Relationship Cardinality

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

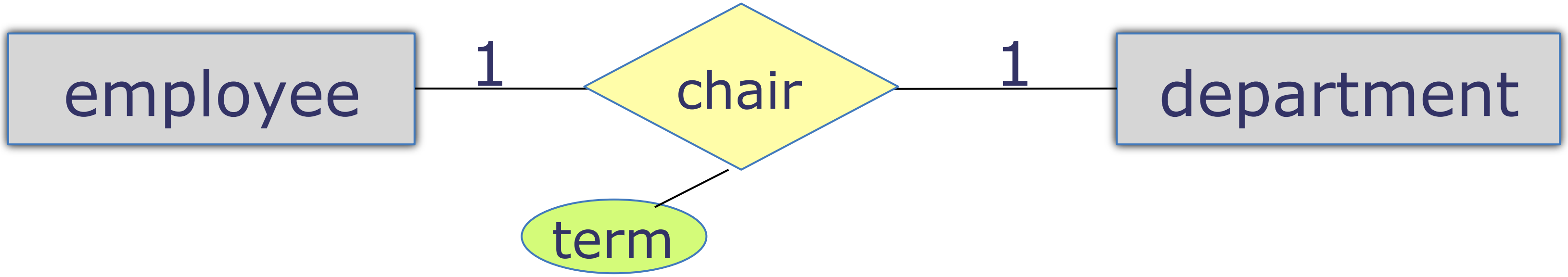


- N:M relationship (not so rare, but try to avoid)

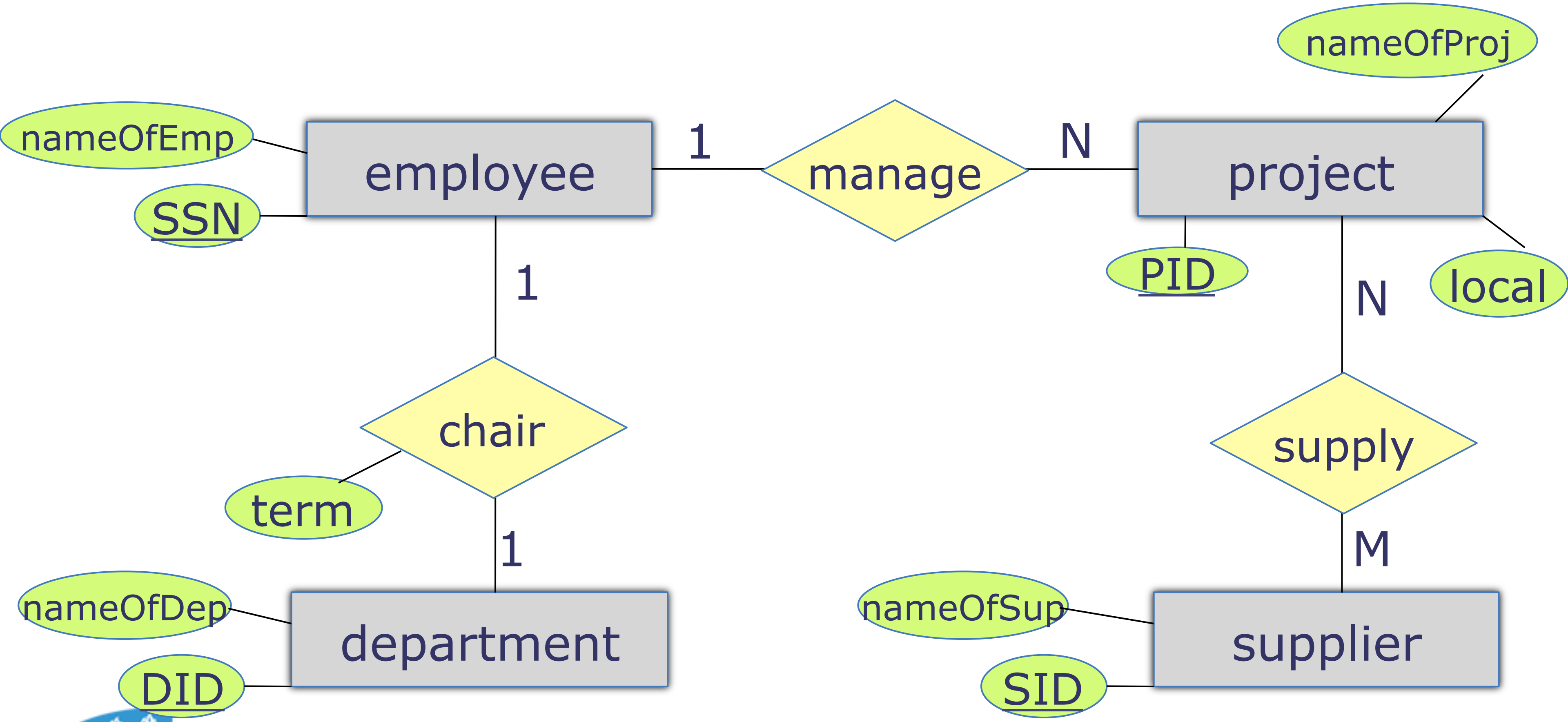


Relationship Attributed

- An attribute of a relationship only exists due to such association

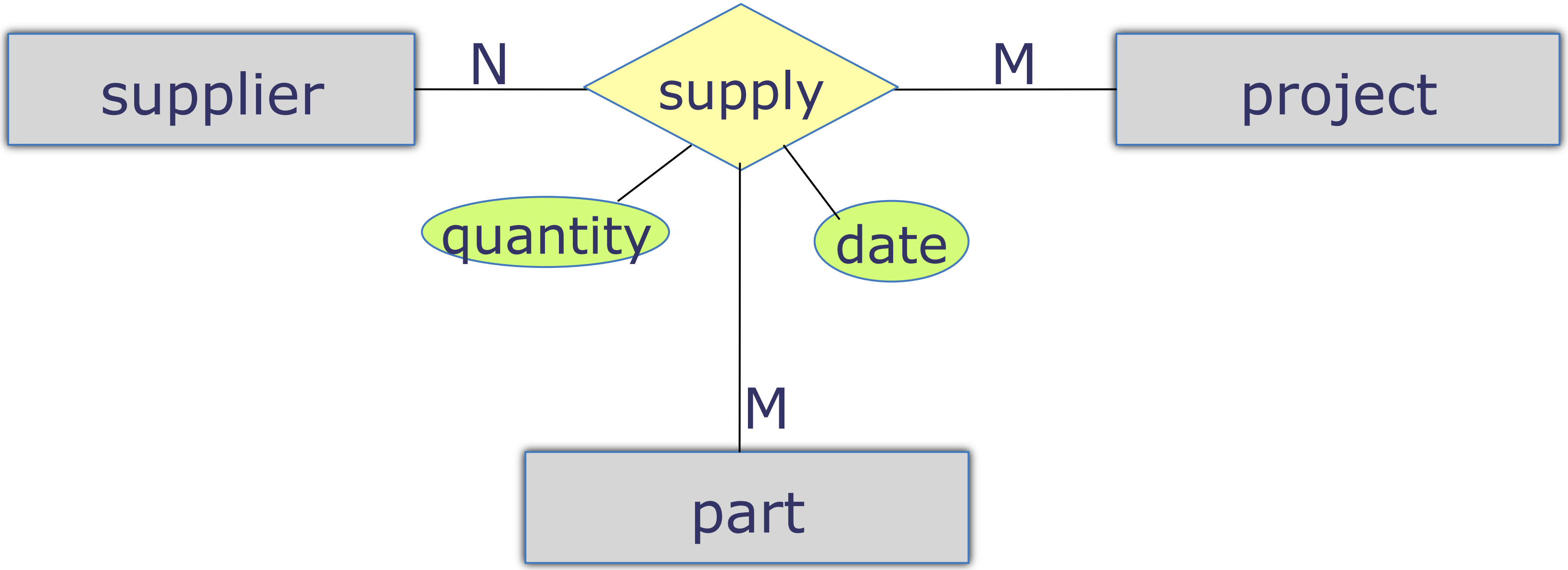


Example


nameOfProj
nameOfEmp
employee
1
manage
N
project
SSN
PID
local
1
N
chair
supply
term
1
M
nameOfDep
department
nameOfSup
supplier
DID
SID

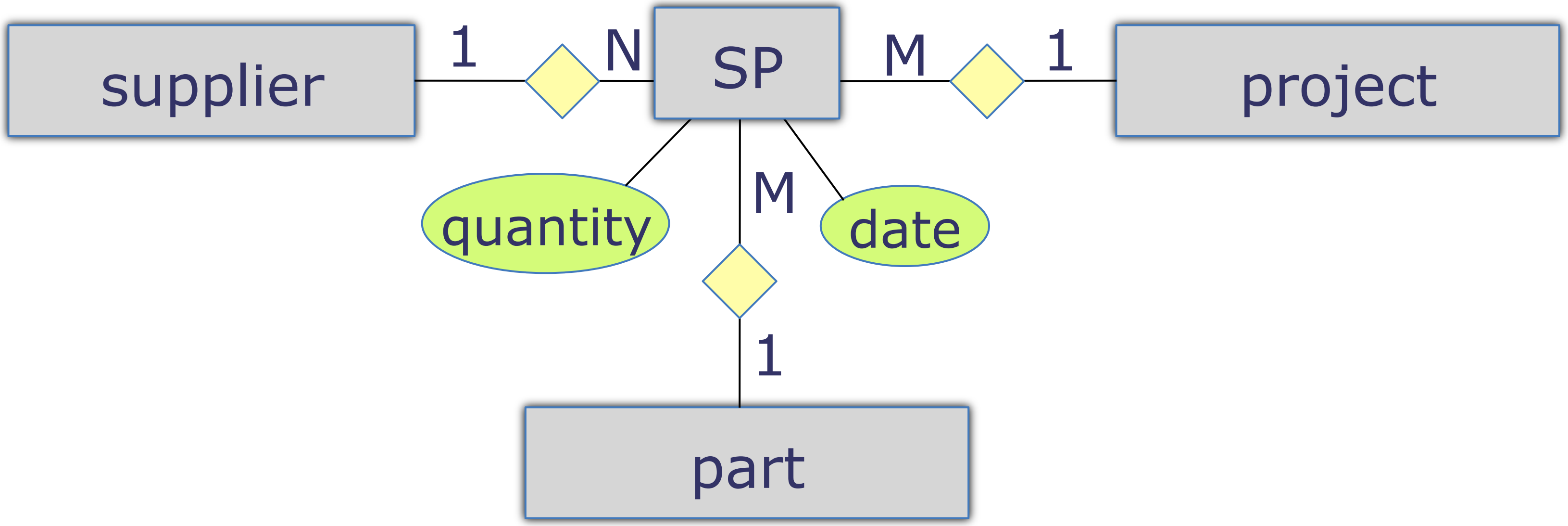
Multi-way Relationships

- Ternary Relationship

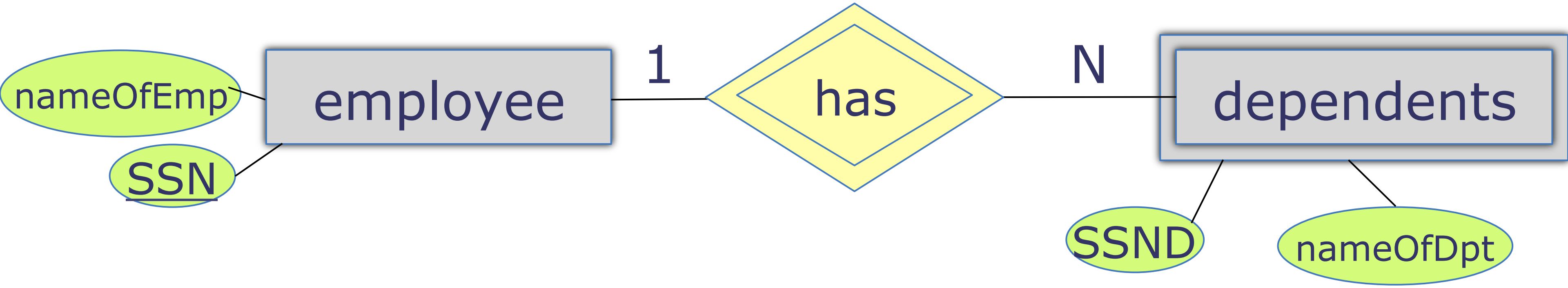


Multi-way Relationships

- Ternary to binary relationship



Weak Entities

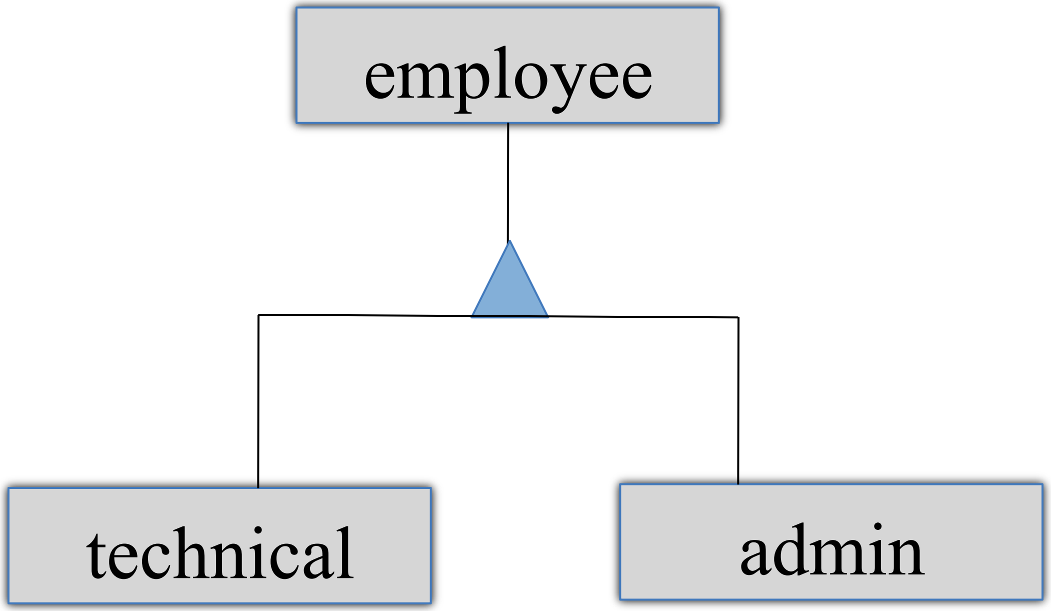


- Cannot be identified by its attributes alone
- Requires a foreign key in conjunction with its attributes

Modeling Hierarchy

Data is naturally hierarchical

(as is the world)



- Not all database systems implement inheritance

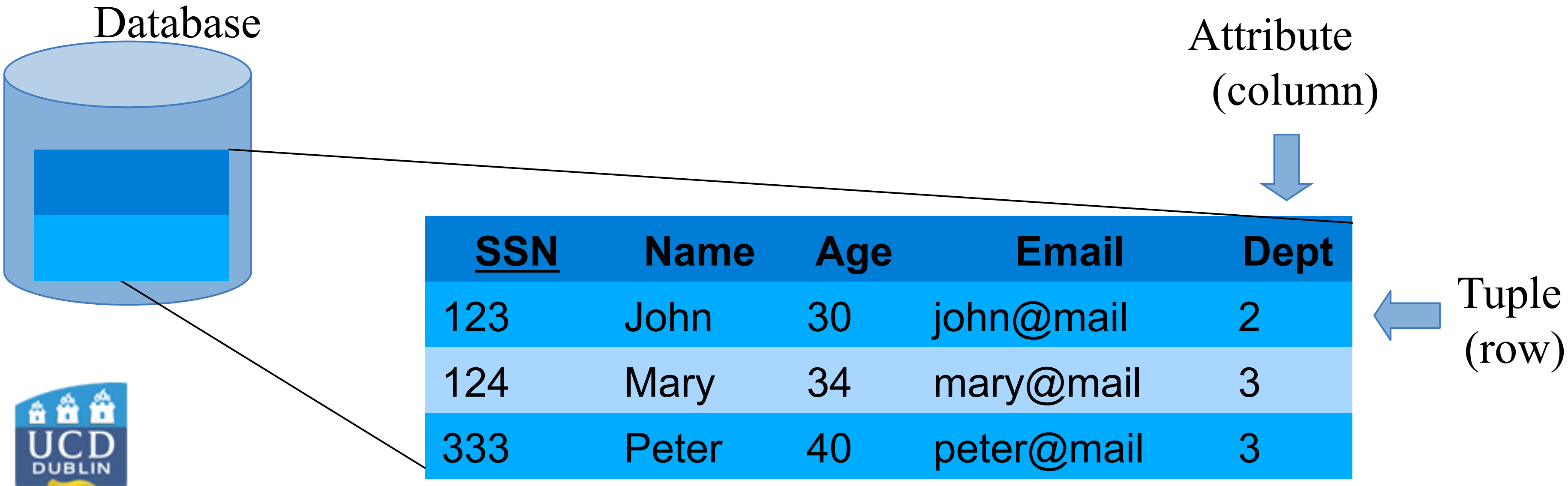
Exercise

1. Entities: Professor, Student, Course
2. Relationships: teach, register
3. Attributes: ???

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

Attributed (Columns)

- Data type:
  - integer, float, string, date/time, binary
- Domain:
  - set of atomic values (e.g., SSN is a set of 9 digits)
  - mono-valued

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 the type of its attribute
  - 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 canno
be identifie

Constraints

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

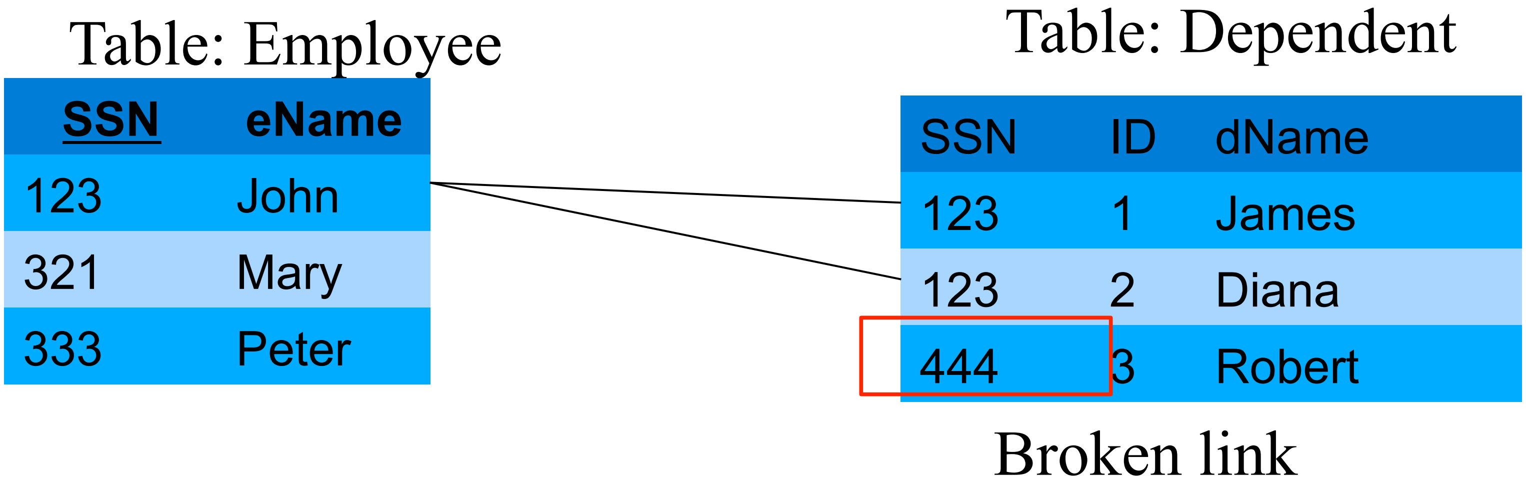


Table: Employee

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 Normalisation

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