Web & Database Technology

More Modelling

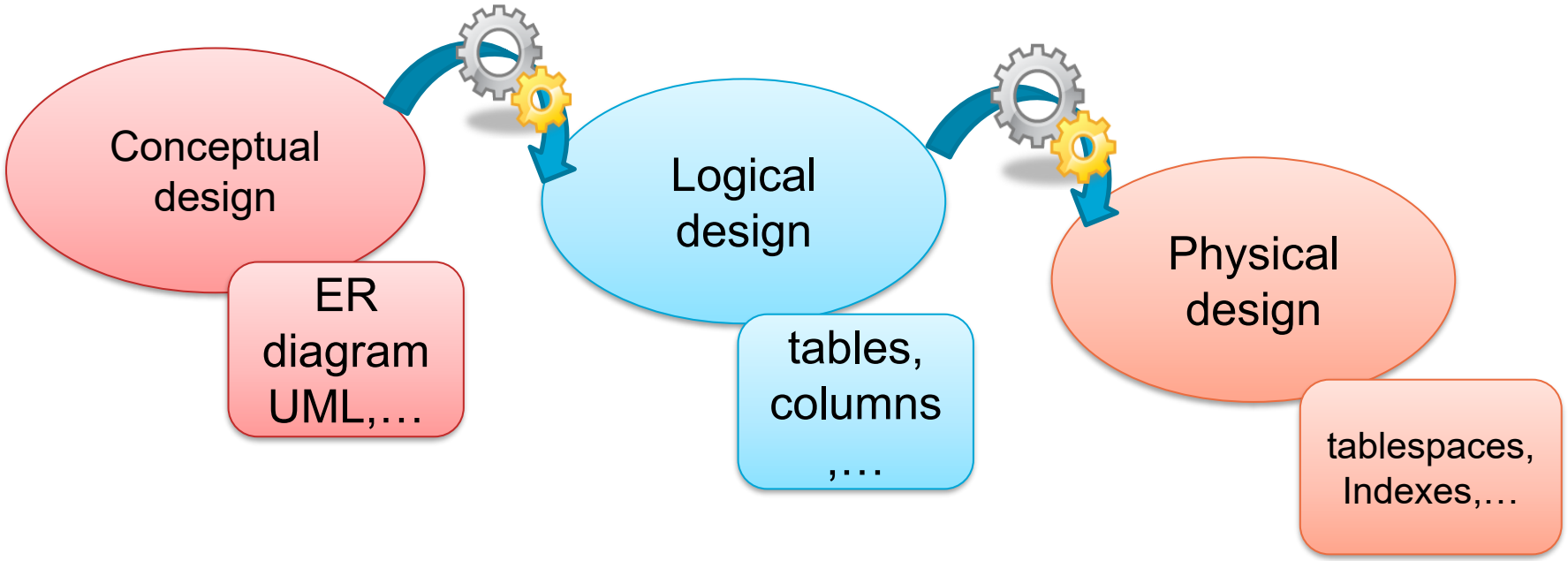
Christoph Lofi

TU Delft WIS Web Information Systems

- Goal of today's **lecture**:
  Let's discuss modelling!
- Goal of today's **tutorial**:
  Let's practice modeling and look into aspects **of (E)ER diagramming**

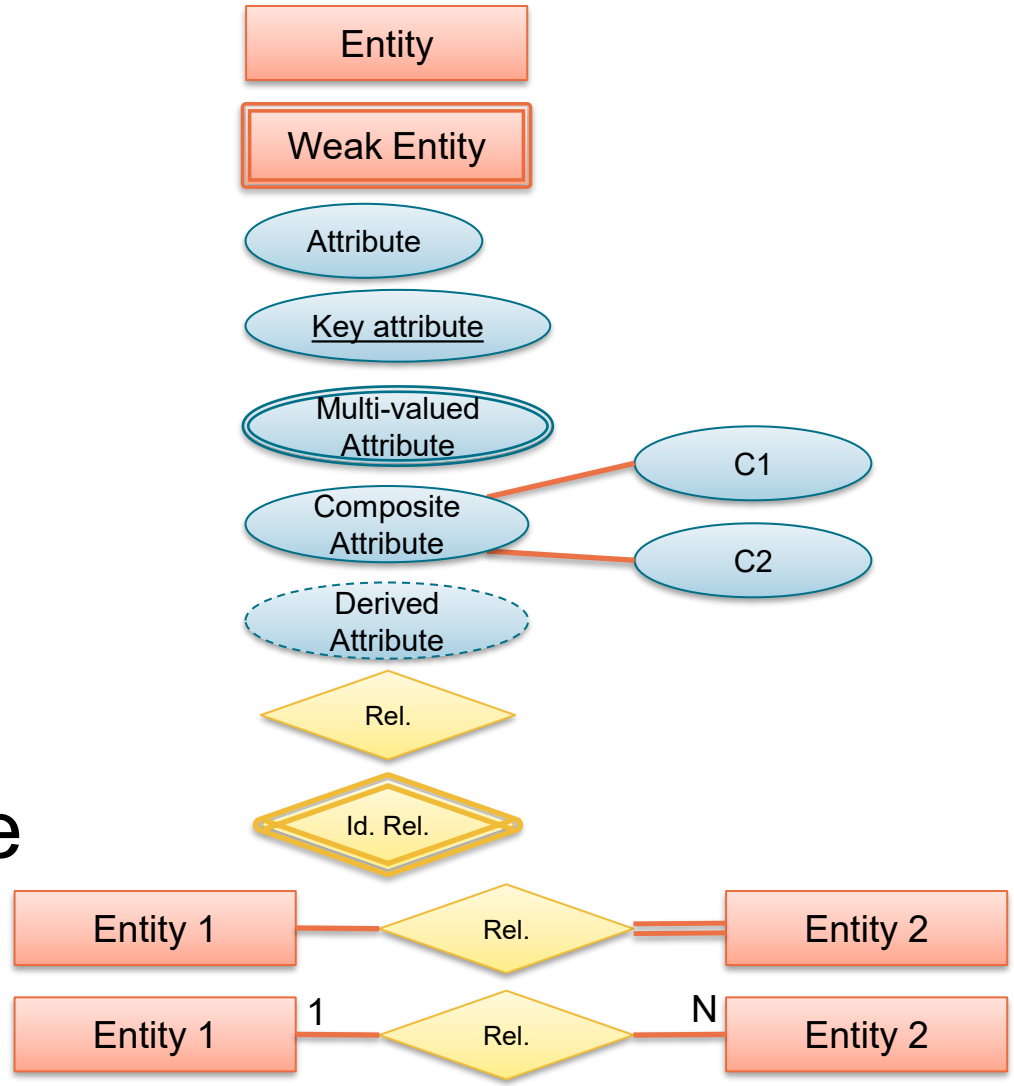
Conversion from ER

- After modeling a conceptual schema (e.g., using an **ER diagram**), the schema can be (semi-) **automatically** transformed into a **relational schema**



Summary ER Modelling

- Entity (Type)
- Weak Entity (Type)
- Attribute
- Key Attribute
- Multi-valued Attribute
- Composite Attribute
- Derived Attribute
- Relationship Type
- Identifying Relationship Type
- Total Participation
- 1:N Cardinality



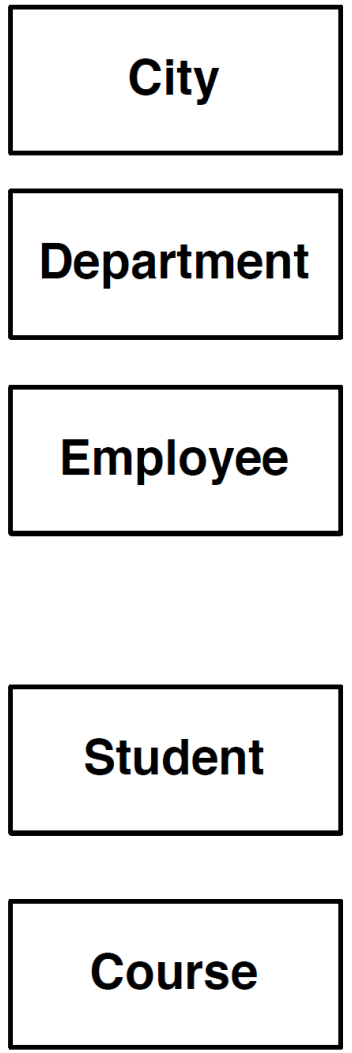
Recap Of EER

(also the tutorial)

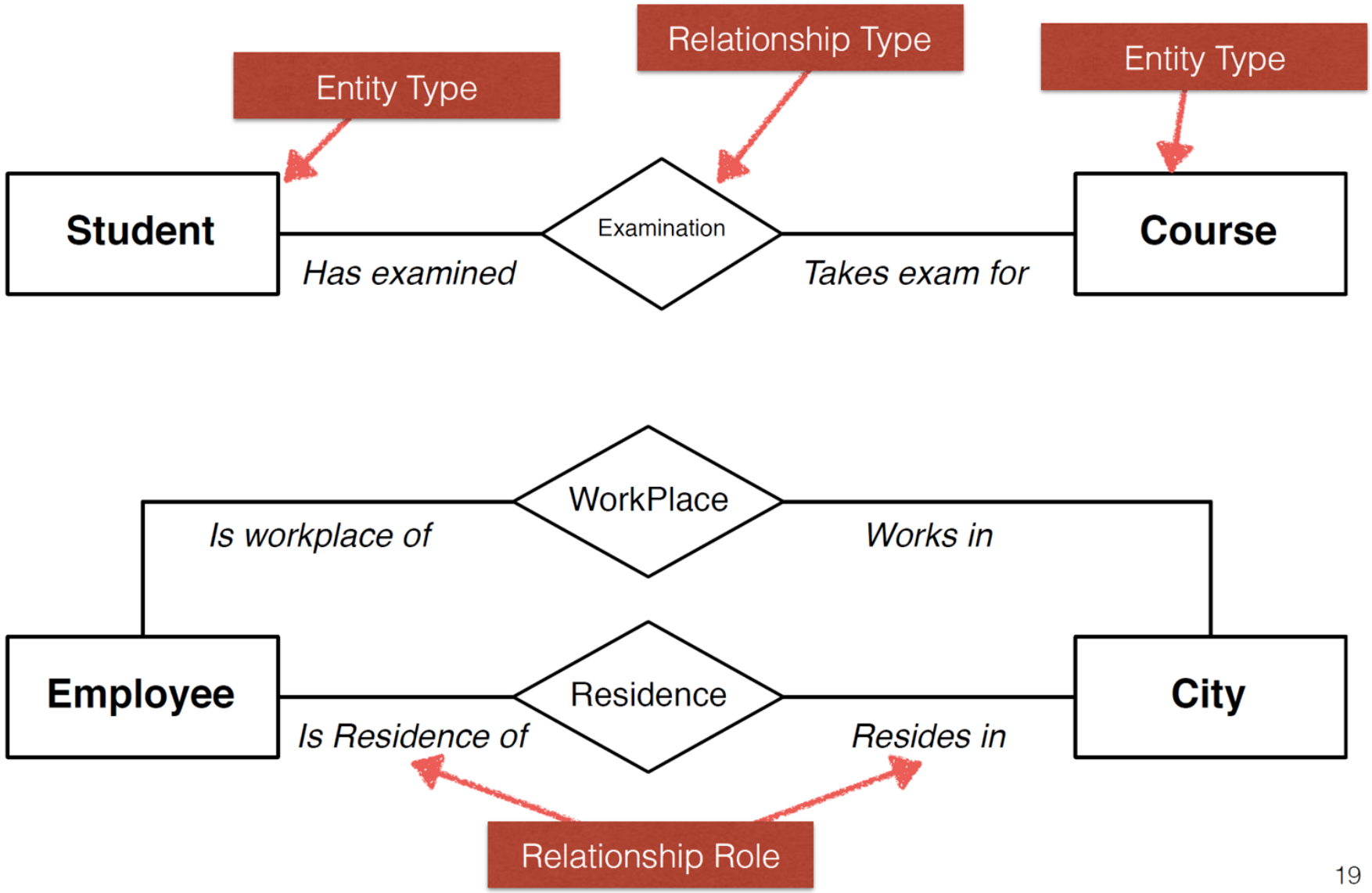
TU Delft WIS Web Information Systems

Entity Types

- Classes of objects (e.g., facts, things, people) having:
  - Common properties
  - Autonomous existence
- Examples:
  - Commercial organization: CITY, DEPARTMENT, EMPLOYEE, PURCHASE, and SALE
  - University: STUDENT, COURSE
- An occurrence of an entity type is an object (or an entity) of the class that the entity type represents.

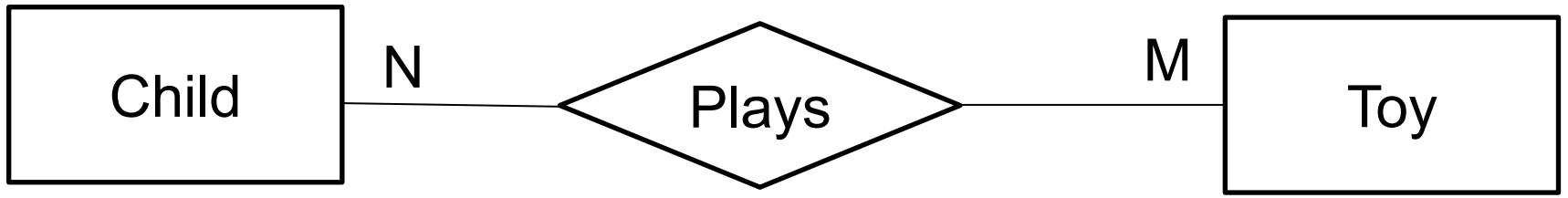


Relationship types in the E-R model



Relationship Types

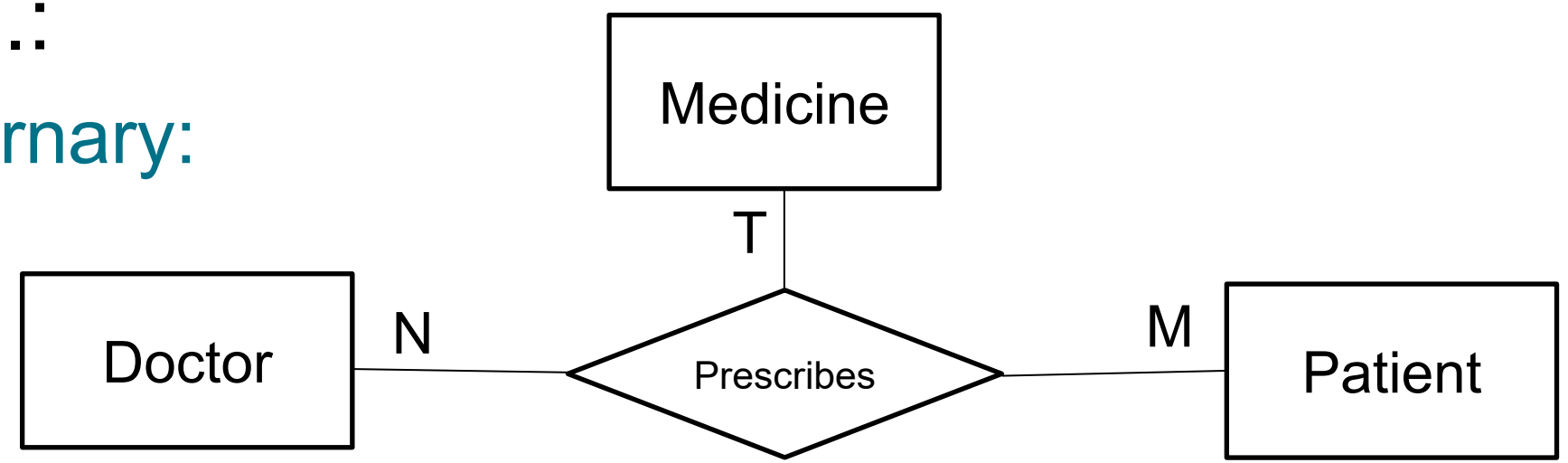
- E.g.:
  - Binary:



- A **Child** *Plays* with M **Toys**, and a **Toy** can be *Played* with by N **Children**

Relationship Types

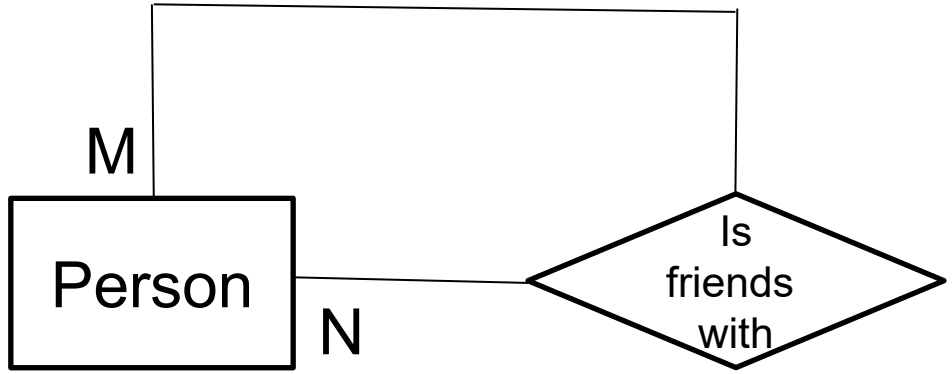
- E.g.:
  - Ternary:



- A **Doctor** can *Prescribe* T **Medicine** to M **Patients**, a **Medicine** can be *Prescribed* by N **Doctors** to M **Patients**, a **Patient** can be *Prescribed* T **Medicine** by N **Doctors**

Relationship Types

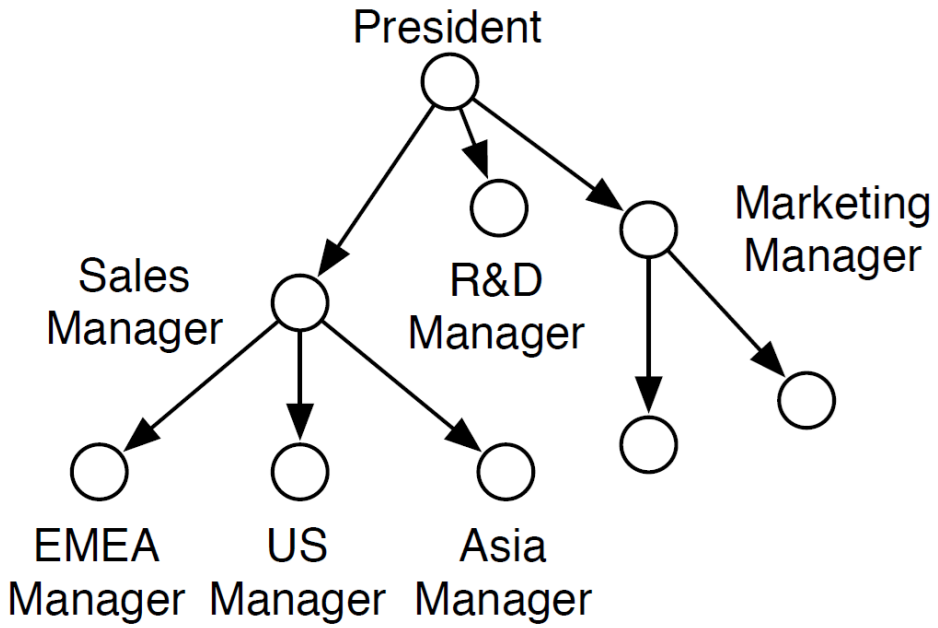
- E.g.:
  - Recursive:

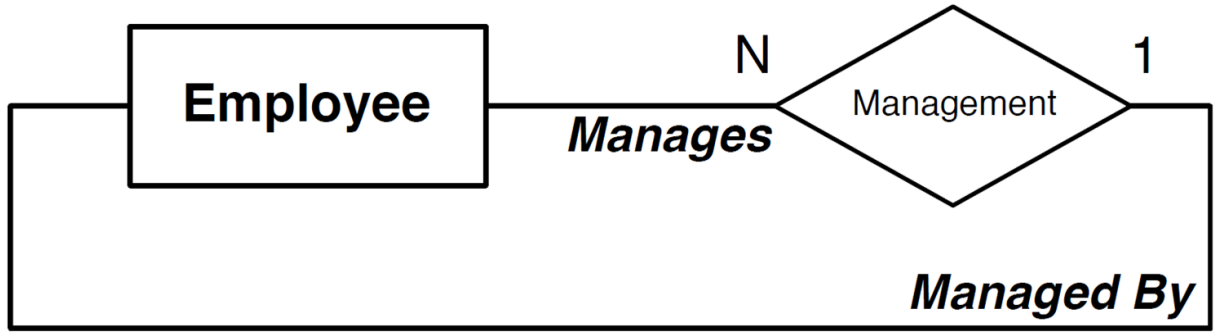


- A **Person** can *Be friends with* M **Persons**
- Can a Person Be friends with themselves?

Recursive Relationship Types

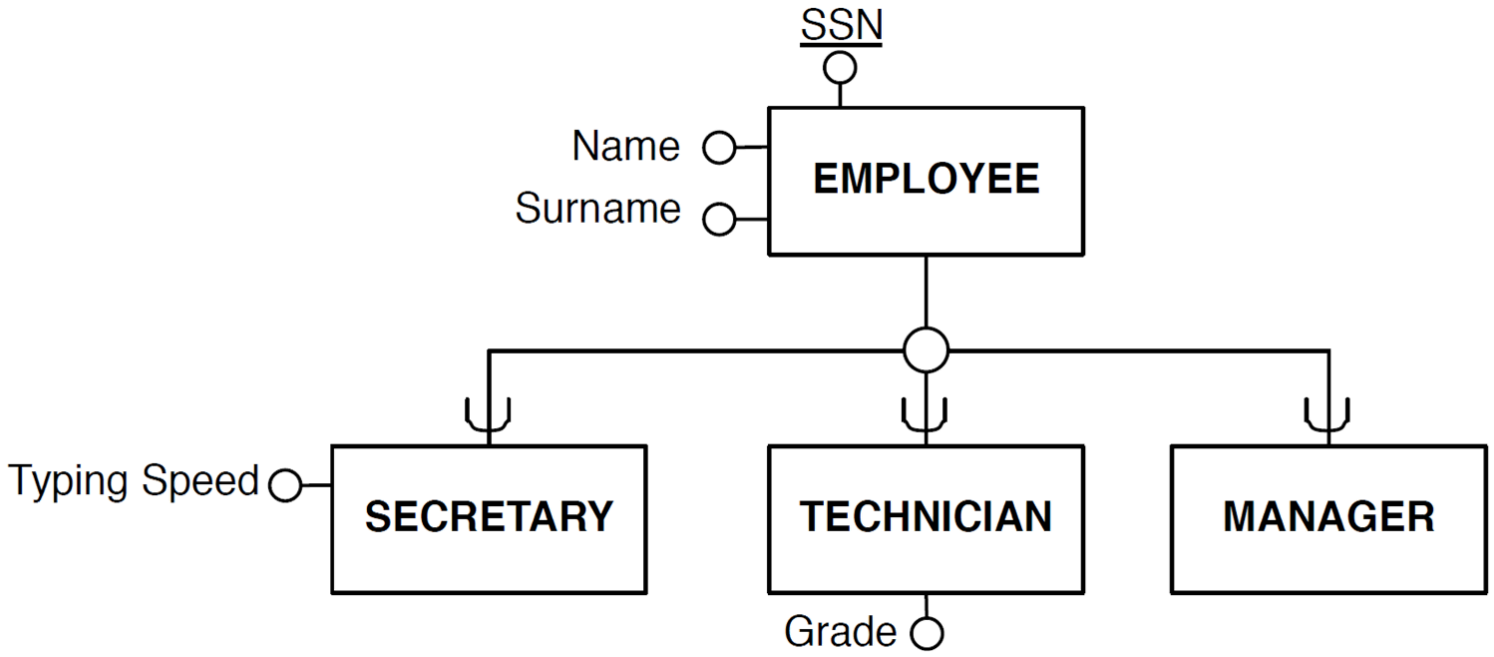
- Relationship where the same entity participates with different roles





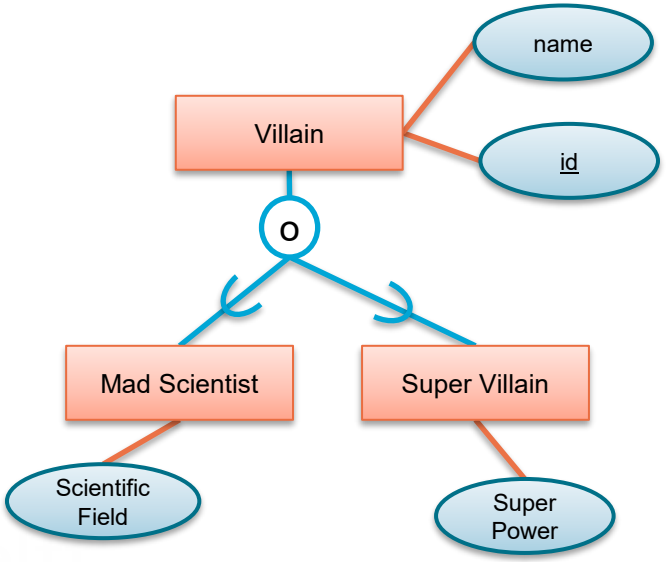
Properties of Specialisation

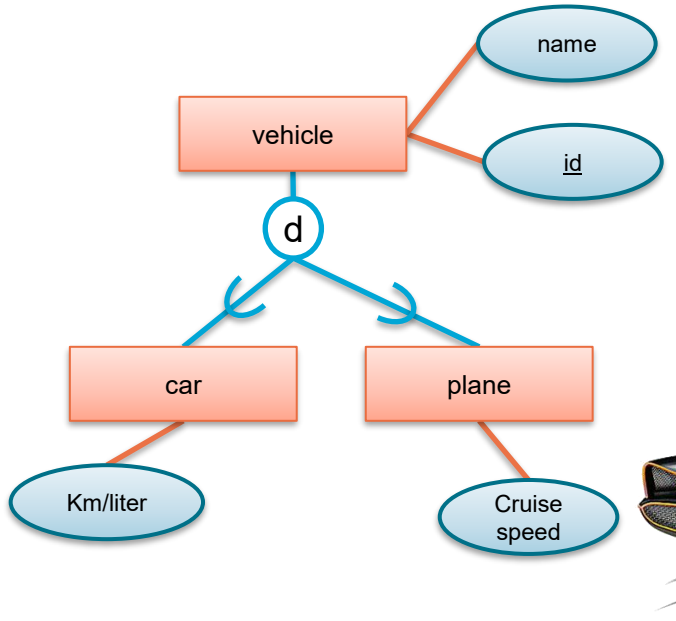
- Every occurrence of a child entity is also an occurrence of the parent entity
- Inheritance
  - Every property of the parent entity (attributes, identifiers, relationships and other generalisations) is also a property of a child entity.
- Specialisation can define specific attributes and/or specific relationship types
  - Specializations can DISJOINT or OVERLAPPING; Partial or Total



Properties of Specialisation

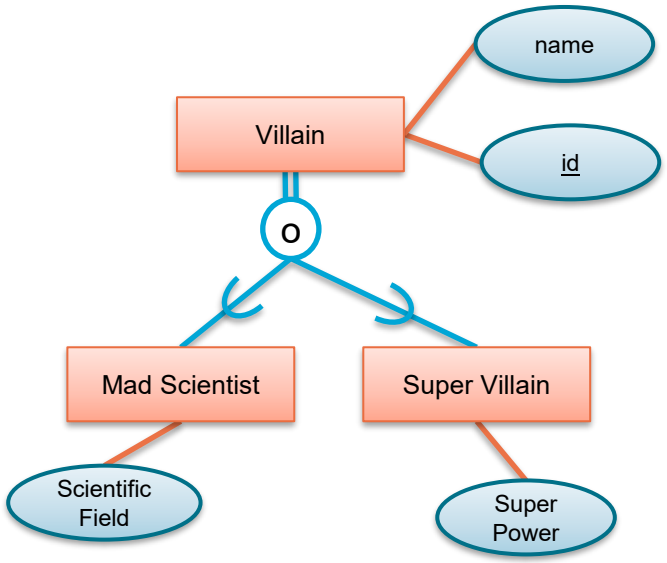
- Supertypes and Subtypes
  - Overlapping and Disjoint
    - Entity can be of multiple subtypes, or of only one





Properties of Specialisation

- Supertypes and Subtypes
  - Total or Partial
    - Total: Each entity must be of one of the subtypes
      - Here: Each villain needs to be a super villain and/or a mad scientist

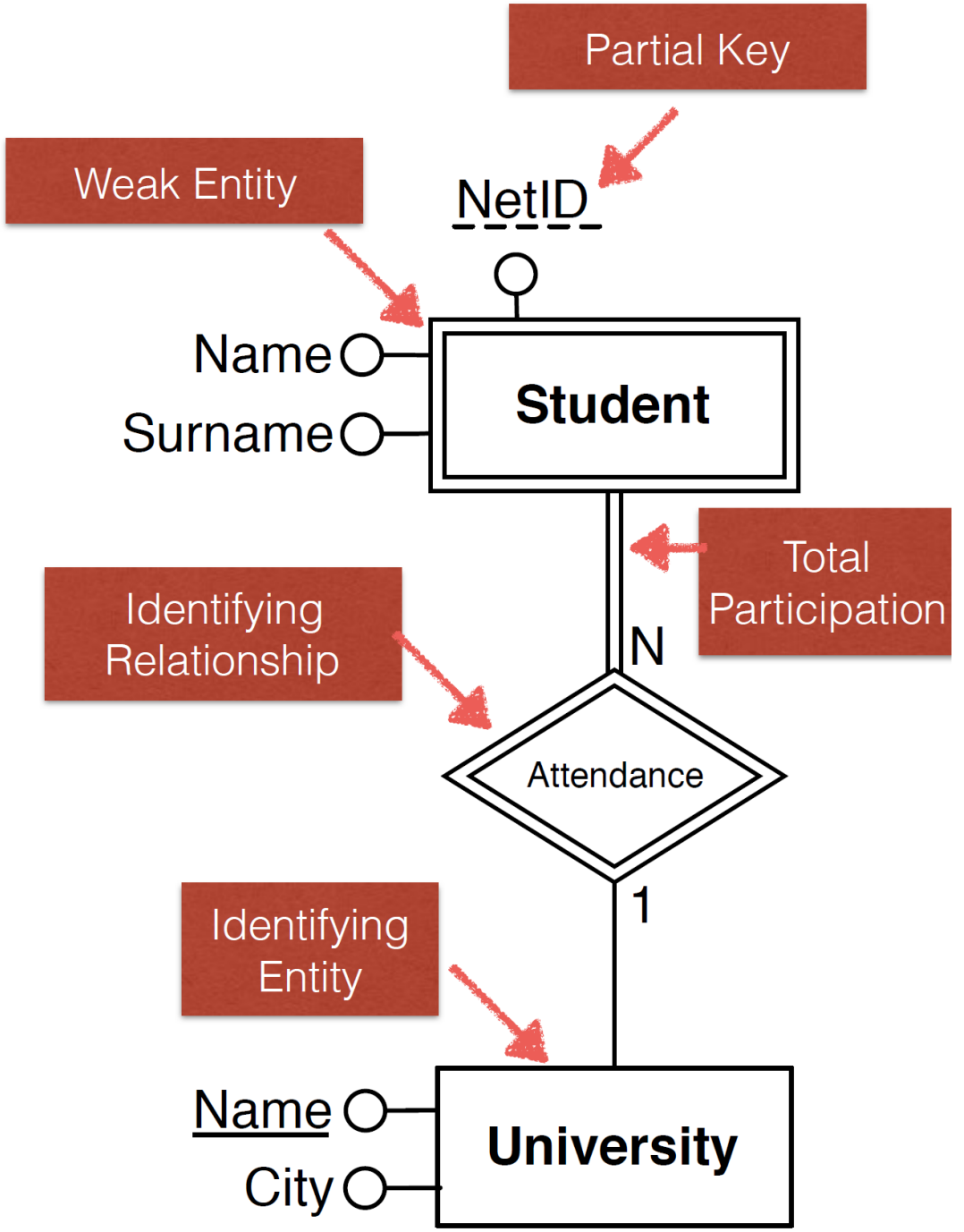


Limitations of EER

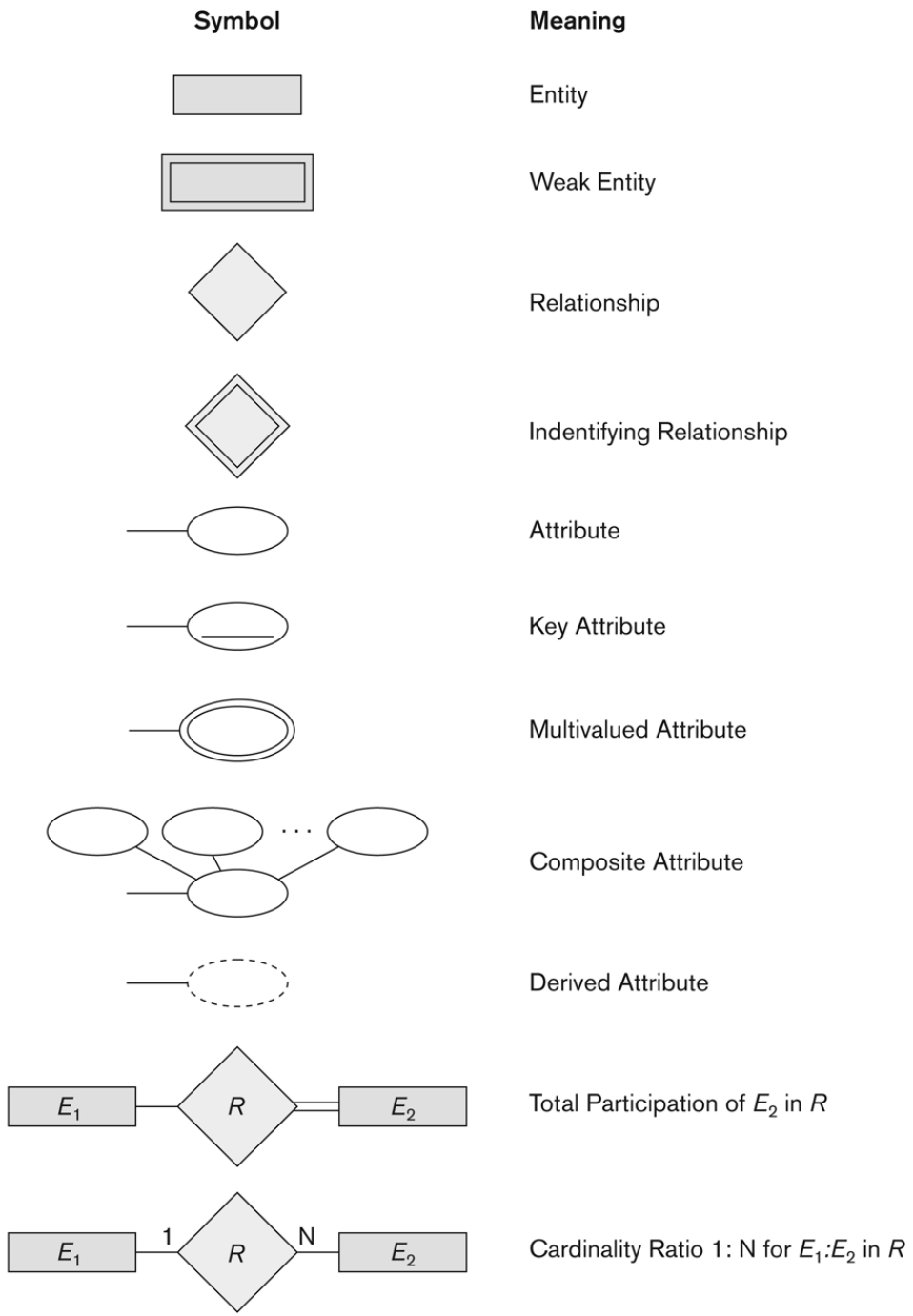
- There are many things (E)ER diagrams can not represent:
  - Arrays
  - Nested structures
  - Reusable Composites
  - Individual entities (i.e., instances)
    - Example before: Christos is an instance of type person, Facebook is an instance of type social media platform, etc.
    - Also, in the burrito examples from the tutorial, each individual burrito did have a database record! (burrito_1, buritto_2, etc.)

Weak Entities

- Sometimes the attributes of an entity are not sufficient to identify its occurrences unambiguously
  - Weak Entities
  - Entities may not have internal identifiers
  - Or partial identifiers (Partial keys)
- Other entities need to be involved in the identification
  - This is called an external identifier
- The relationship that relates a weak entity to its owner is called identifying relationship



Summary ER


Symbol
Meaning
Entity
Weak Entity
Relationship
Indentifying Relationship
Attribute
Key Attribute
Multivalued Attribute
Composite Attribute
Derived Attribute
E_1
R
E_2
Total Participation of E_2 in R
E_1
1
R
N
E_2
Cardinality Ratio 1: N for E_1:E_2 in R

How Weak is your Entity?

TU Delft WIS Web Information Systems

Discussion: Weak Entities

- Weak Entities are typically entities whose **identity** depends on another entity
  - Often, because the weak entity is a part of the strong one, or gets aggregated into the strong one
  - Often, because the weak entity cannot exist without a strong entity
  - Weak entities have partial keys
    - The other part comes from the strong entity

Discussion: Weak Entities

- Is STUDENT a weak entity?

Discussion: Weak Entities

- Is STUDENT a weak entity?
  - Depends on your point of view…
  - WITHIN a university, students are certainly strong entities
    - See systems like Brightspace, MyTuDelft, our Admission System, Queue
  - However, nationally, students are a weak entity, borrowing part of their identity from a “strong” university
  - Which one is “correct”?
    - ….it depends…

Discussion: Weak Entities

- Is “RAILROAD CAR” a weak entity?

Discussion: Weak Entities

- Is “RAILROAD CAR” a weak entity?
  - Again, depends on your viewpoint…
    - Yes, if you look at it from a timetable perspective, communicating to passengers
    - The car is part of the train
    - “This is car 12 of IC 2021”
      - The same car could be car 15 in IC 2022, when serving a different route….

Discussion: Weak Entities

- Is “RAILROAD CAR” a weak entity?
  - Again, depends on your viewpoint…
    - No, if you look at it from a maintenance/logistics perspective
      - “This is CBTX784948”
      - These cars are frequently attached and detached from trains, and cars in one train might go to different destinations…

Discussion: Weak Entities

- Is PATIENT a weak entity?
  - Similar to student: depends on scope…
- Is ROOM a weak entity?
- Is BUILDING a weak entity?
- Is CAR a weak entity?
  - …haha, yes, this gets fun…

Discussion: Weak Entities

- Cars in the Netherlands

Vehicle registration plates of the Netherlands

Article Talk

From Wikipedia, the free encyclopedia

Vehicle registration plates of the Netherlands are vehicle registration plates issued by the RDW (Dutch Vehicle Authority) [nl].[1]

RDW vehicle registration plates are assigned bearing the same "number" which is a sequence of characters composed of letters and digits as that is shown on the vehicle's registration document. The numbering scheme used bears no relation to the place of a vehicle's registration or ownership, and numbers – which are issued in strict time order – identify the vehicle, not its owner. Thus, if a vehicle changes ownership, the registration number remains unchanged.

If the car is registered before January 1st 1978, it may have a dark-blue number plate also called 'historical plates'.[2] Taxis will have a light-blue number plate because they pay a different amount of tax to let people into the car legally. If a taxi does not have a blue number plate on it, it is an illegal taxi and the driver will charge a lesser fare to the person in the car.

Discussion: Weak Entities

- Cars

print download

VOLKSWAGEN GOLF, uit november 2004

Dit voertuig had destijds een nieuwprijs vanaf **€19.995,-** en is een origineel Nederlandse auto.

De auto kan van 0-100km/h in **11,5 seconden** en heeft een topsnelheid van **192 km/h** en een maximum vermogen van **105 PK**.

Deze GOLF verbruikt gemiddeld **1 liter op de 14.3 kilometer**. Met een **CO2-uitstoot van 168 gram** per kilometer krijgt deze VOLKSWAGEN een **zuinigheidslabel C**.

NL 43-PS-PH

rapportdatum: 18-11-2024

Algemene informatie

Merk	VOLKSWAGEN
Type	GOLF Basis
Kleur	grijs (reflexzilver metallic)
Bouwjaar	10-11-2004
Nieuwprijs	€ 19.995
Brandstof	Benzine
Vermogen	77 kW / 105 PK bij 5.700 rpm
0-100km/h	11,5 seconden
Topsnelheid	192 km/h
Transmissie	5 v., handgeschakeld
Aantal eigenaren	(bekijk uitgebreid rapport)
Importauto	Nee
APK tot	01-12-2024

e Netherlands

icle registration plates issued

e same "number" which is a
s that is shown on the vehicle's
rs no relation to the place of a
are issued in strict time order
anges ownership, the

ave a dark-blue number plate
e number plate because they
gally. If a taxi does not have a
will charge a lesser fare to the

https://www.kentekencheck.nl/volkswagen/43-PS-PH

Discussion: Weak Entities

- Cars

print download

VOLKSWAGEN GOLF, uit november 2004

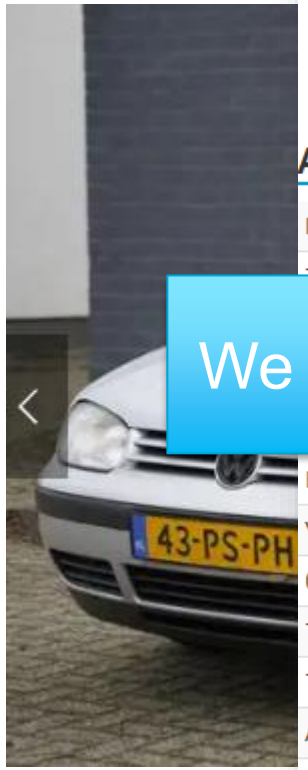
Dit voertuig had destijds een nieuwprijs vanaf **€19.995,-** en is een origineel Nederlandse auto.

De auto kan van 0-100km/h in **11,5 seconden** en heeft een topsnelheid van **192 km/h** en een maximum vermogen van **105 PK**.

Deze GOLF verbruikt gemiddeld **1 liter op de 14.3 kilometer**. Met een **CO2-uitstoot van 168 gram** per kilometer krijgt deze VOLKSWAGEN een **zuinigheidslabel C**.

NL 43-PS-PH

rapportdatum: 18-11-2024



Algemene informatie

Merk	VOLKSWAGEN
Brandstof	Benzine
Vermogen	77 kW / 105 PK bij 5.700 rpm
0-100km/h	11,5 seconden
Topsnelheid	192 km/h
Transmissie	5 v., handgeschakeld
Aantal eigenaren	(bekijk uitgebreid rapport)
Importauto	Nee
APK tot	01-12-2024

We could argue that a Dutch car is a Strong Entity

e Netherlands

icle registration plates issued

e same "number" which is a
s that is shown on the vehicle's
rs no relation to the place of a
are issued in strict time order
anges ownership, the

ave a dark-blue number plate
e number plate because they
gally. If a taxi does not have a
will charge a lesser fare to the

https://www.kentekencheck.nl/volkswagen/43-PS-PH

Discussion: Weak Entities

1-letter area code Z for Zwickau

2-letter area code TR for Trier

3-letter area code CUX for Cuxhaven

Vehicle registration plates of Germany

Article Talk

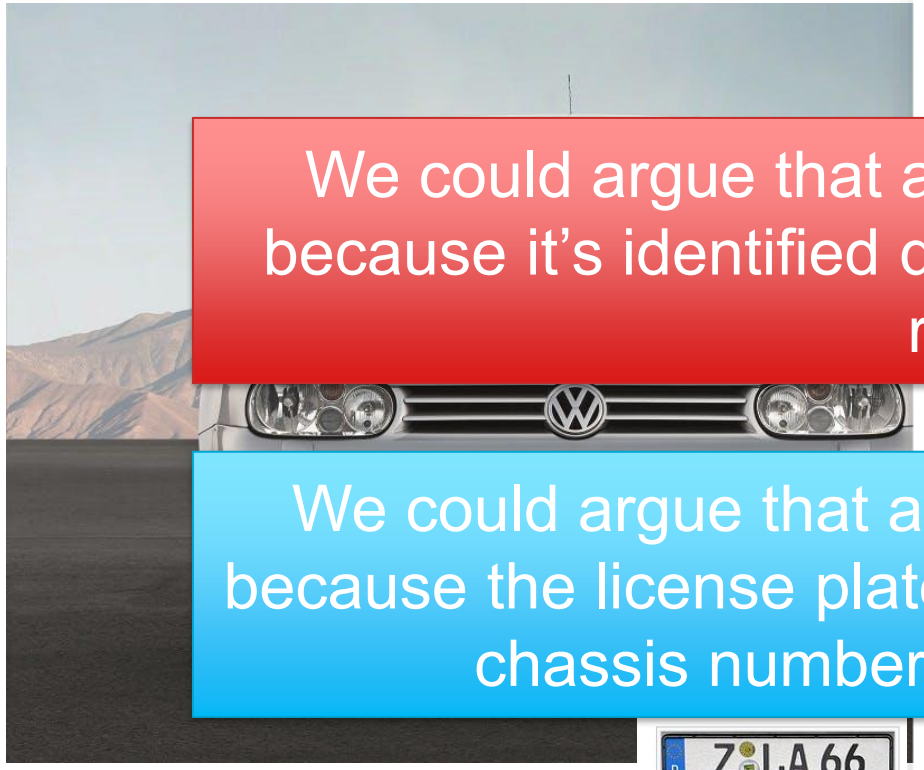
From Wikipedia, the free encyclopedia

Vehicle registration plates (German: *Kraftfahrzeug-Kennzeichen* or, more colloquially, *Nummernschilder*) are mandatory alphanumeric plates used to display the *registration mark* of a vehicle registered in Germany. They have existed in the country since 1906, with the current system in use since 1956. German registration plates are alphanumeric plates in a standardised format, issued officially by the district authorities.[1][2]

All motorised vehicles participating in road traffic on public space, whether moving or stationary, have to bear the plates allotted to them, displayed at the appropriate spaces at the front and rear. Additionally, the official seals on the plates show their validity which can also be proven by the documentation coming with them. Motorcycles and trailers carry only a rear plate.

A significant feature of German vehicle registration plates is the area code, which can be used to tell the district of registration. It has developed into a widespread habit in Germany, even a children's game when travelling, to guess "where that vehicle is from".[3][4]

Discussion: Weak Entities



We could argue that a German car is a Weak Entity, because it's identified depends on the owner's place of residence

We could argue that a German car is a Strong Entity, because the license plate is actually meaningless and the chassis number is what gives identity….

Vehicle registration plates of Germany

…n or, more colloquially,
…splay the *registration*
mark of a vehicle registered in Germany. They have existed in the country since 1906,
with the current system in use since 1956. German registration plates are alphanumeric
…horities.[1][2]

…whether moving or
…e appropriate spaces
…how their validity which
can also be proven by the documentation coming with them. Motorcycles and trailers
carry only a rear plate.

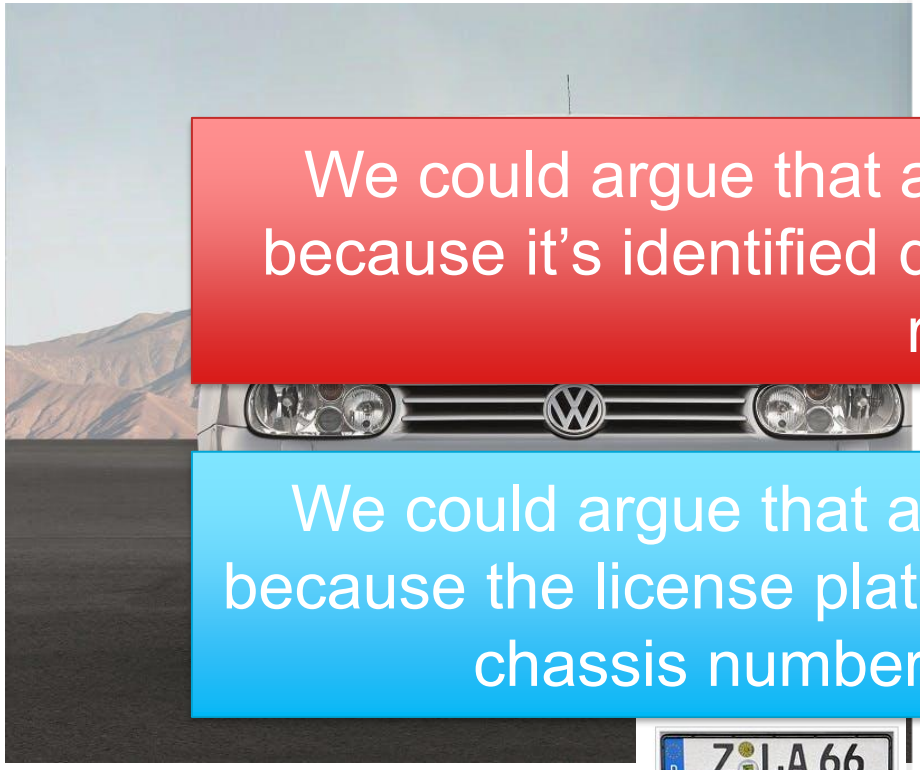
A significant feature of German vehi
be used to tell the district of registrat
Germany, even a children's game wh
from".[3][4]

1-letter area code Z for Zwickau

2-letter area code TR for Trier

3-letter area code CUX for Cuxhaven

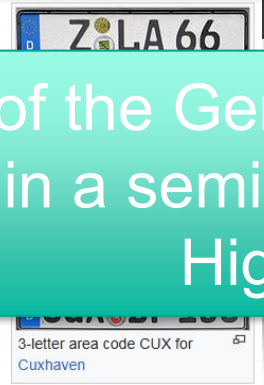
Discussion: Weak Entities



Vehicle registration plates of Germany

…n or, more colloquially,
…splay the *registration*
mark of a vehicle registered in Germany. They have existed in the country since 1906,
with the current system in use since 1956. German registration plates are alphanumeric
…horities.[1][2]

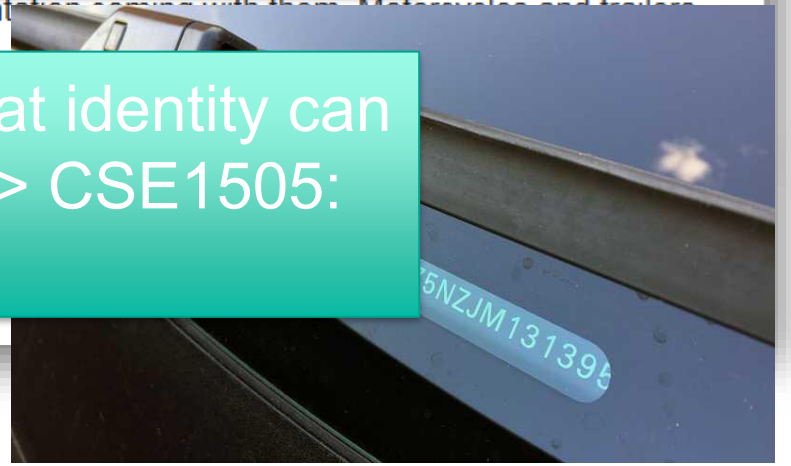
…whether moving or
…e appropriate spaces
…how their validity which
can also be proven by the documentation coming with them. Motorcycles and trailers



We could argue that a German car is a Weak Entity, because it's identified depends on the owner's place of residence

We could argue that a German car is a Strong Entity, because the license plate is actually meaningless and the chassis number is what gives identity….

An advantage of the German approach is that identity can be managed in a semi-districted fashion (-> CSE1505: High-Low keys)



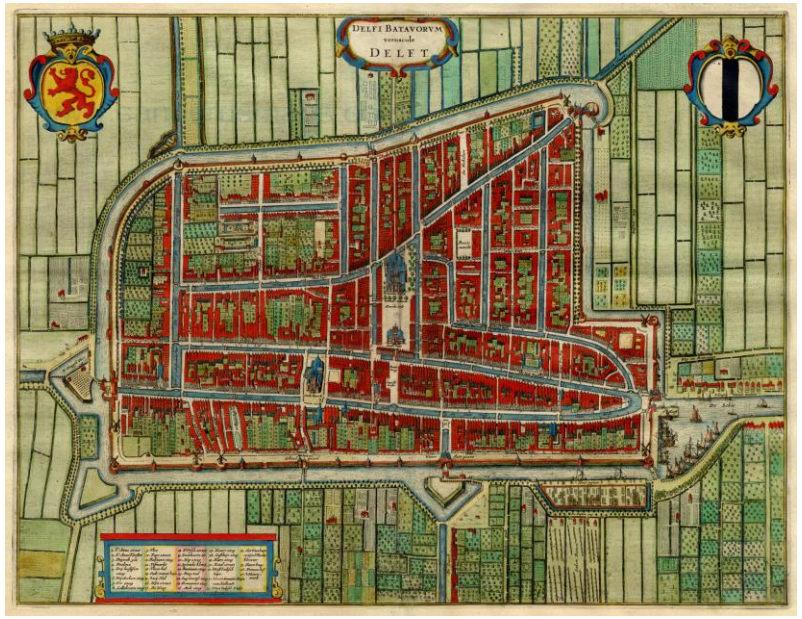
Keys and Identifying Things

The issue with database keys and how this can mess up a whole country.

TU Delft WIS Web Information Systems

Identity and Modelling

- Imagine: In the early Renaissance Age you want to develop an address system for a new postal service…
  - What identifies a mail delivery location?
  - How do we make addresses?



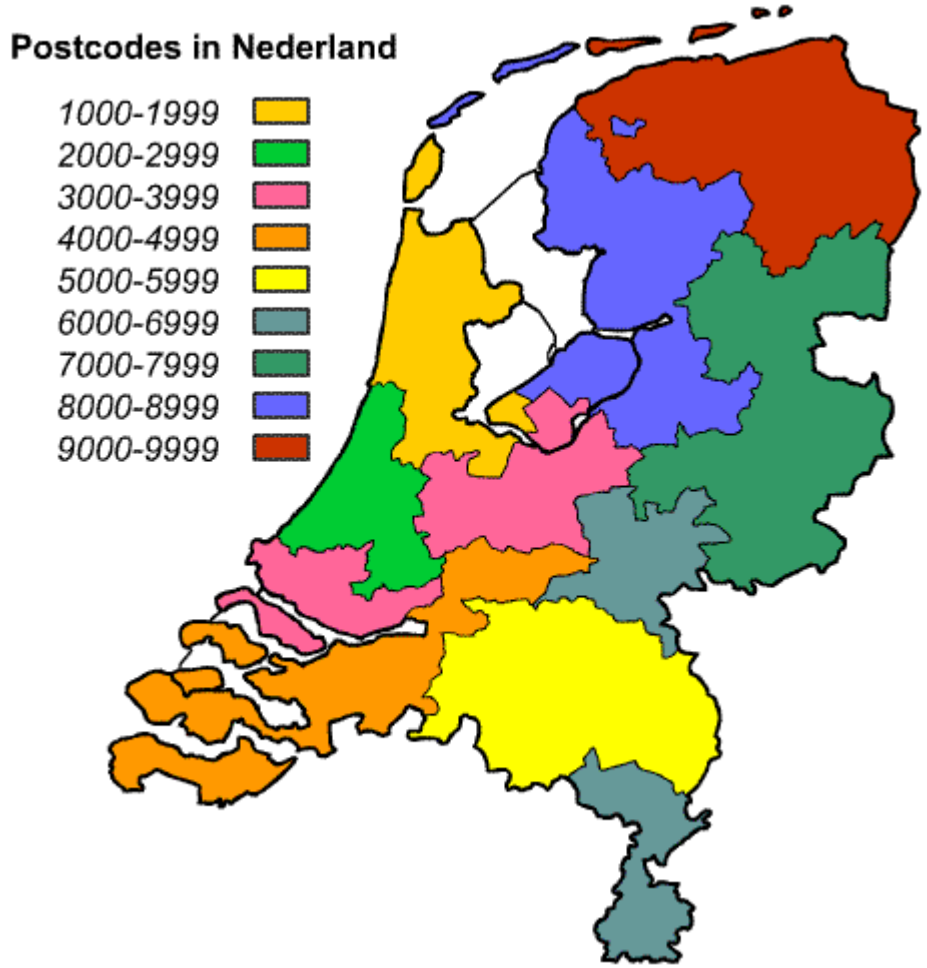
  - https://en.wikipedia.org/wiki/Cursus_publicus
  - https://en.wikipedia.org/wiki/Thurn-und-Taxis_Post

Later in WDT: Functional Dependencies

- The relational schema requires a **primary key** for each relation/table
  - A primary key **identifies** the tuple
  - **Primary Keys** are (manually) selected from Candidate Keys
  - **Candidate Keys** are the results of Functional Dependencies
    - i.e.: which attributes define others
  - To some extent, functional dependencies can be designed by us
    - So, design question: “What defines things?”, “What implies other things?”

DETOUR

Addresses in NL



Postcodes in Nederland

Artikel Overleg Lezen Bewerken Brontekst bewe

Uit Wikipedia, de vrije encyclopedie

Postcodes in Europees **Nederland** worden door PostNL aan door gemeenten gegenereerde adresgegevens toegekend.

In 1977 is de postcode ingevoerd door vroegere PTT in Nederland. PTT had een postcodegids uitgereikt.

De Nederlandse postcode bestaat uit vier cijfers, een spatie en twee letters. Dit geeft een postcode de vorm: 1234_AB. De cijfers geven de stad, het dorp, de buurt of wijk aan, de letters zijn specifieker en geven de straat of een deel daarvan aan.

Een postcodegebied (de vier cijfers) ligt nagenoeg altijd binnen één gemeente, dus als een dorp of buurtschap op een gemeentegrens ligt, heeft het gewoonlijk meer dan één postcode. Het komt slechts zeer zelden voor dat twee plaatsnamen de cijfers van hun postcode gemeen hebben. In enkele gevallen, bijvoorbeeld door grenswijzigingen van dorpen of gemeenten sluit de praktijk niet meer naadloos aan bij deze beoogde eenduidigheid. Om praktische redenen worden dan vaak bestaande postcodes gehandhaafd. Er zijn overigens ook cijfercombinaties die maar één straat aangeven.

Soms komen er binnen één postcode twee straatnamen voor, maar dan altijd zo dat de huisnummers verschillen.

Nederlandse adressen volgen het volgende stramien:

naam
straatnaam huisnummer
postcode (twee spaties) PLAATSNAAM

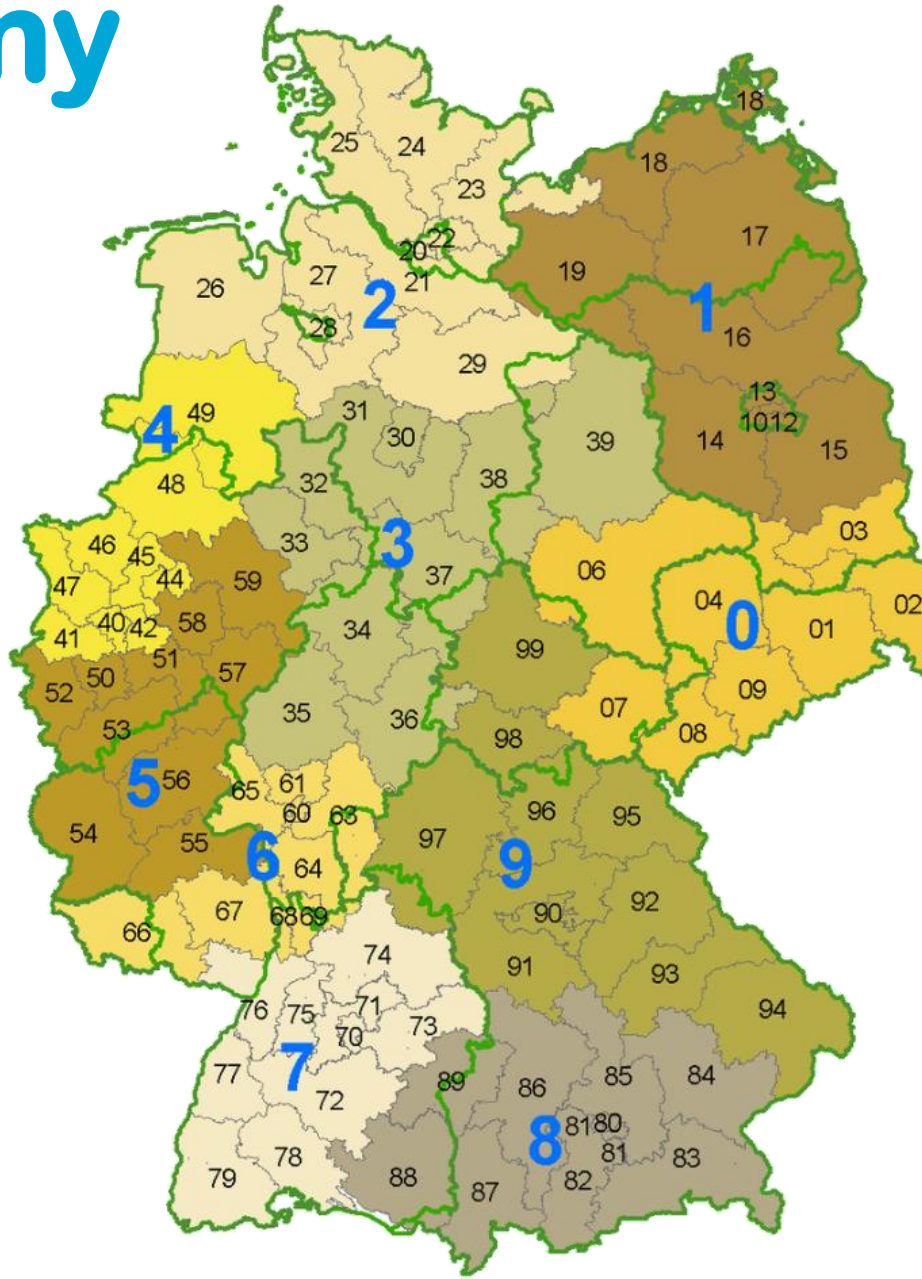
dus bijvoorbeeld

Stadhuis van Zwolle
Grote Kerkplein 15
8011 PK ZWOLLE

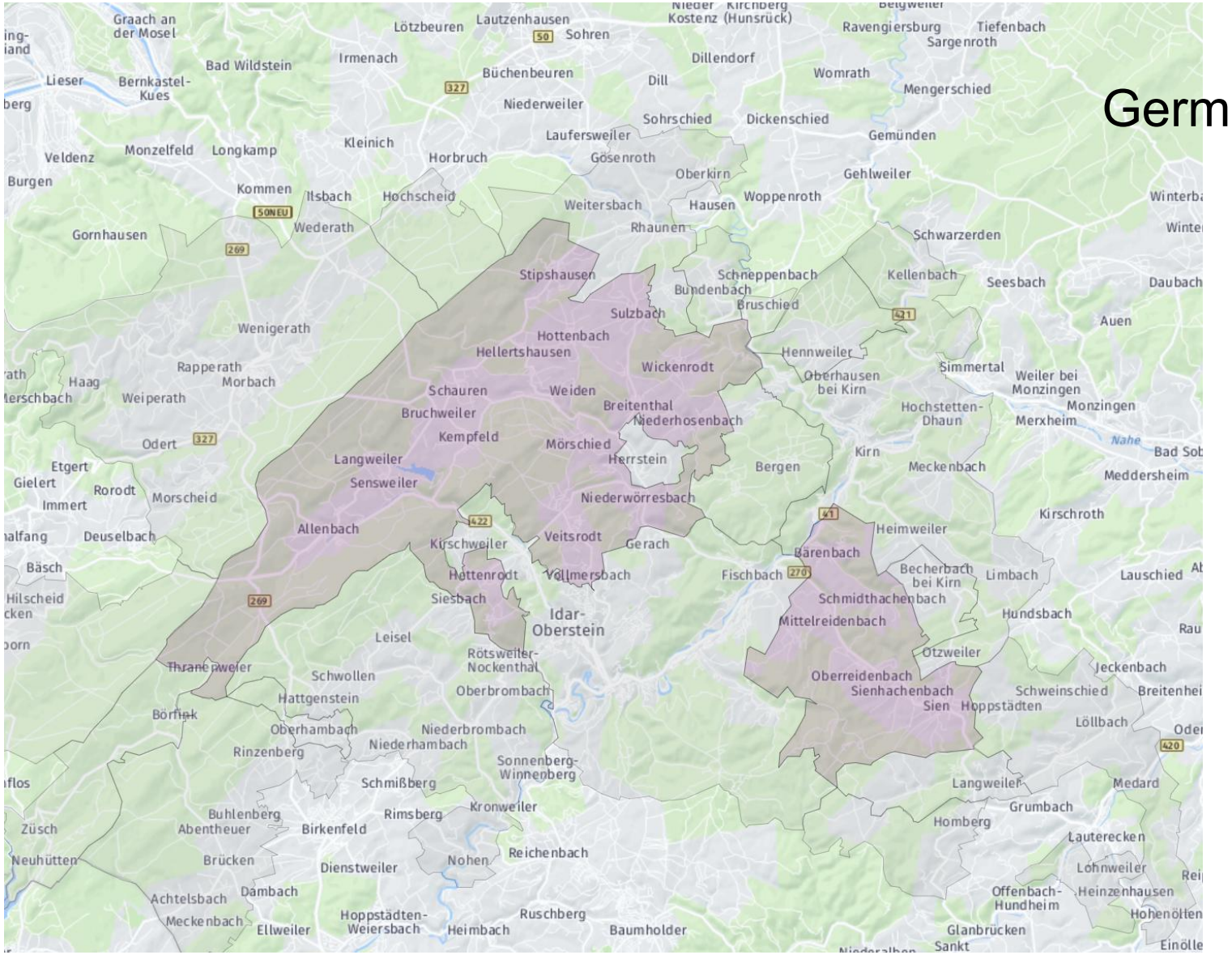
De combinatie van postcode en huisnummer is uniek. In beginsel is voor adressering van post dus niet vereist dat de plaatsnaam of de straatnaam worden vermeld, al is dit wel zo makkelijk voor de postbezorgers. Deze eigenschap is bijzonder: lang niet alle postcodestelsels zijn zo gedetailleerd als het Nederlandse. In veel andere landen (bijvoorbeeld België, Frankrijk en Zwitserland) geeft een postcode alleen maar een stad

Addresses in Germany

- Postcode 5 digits
  - First 3 digits are the general region
  - Postcode is NOT unique per city/settlement
- Street Name
  - Street name is typically (but not always) unique per settlement, but per post code
- House Number
  - Only unique per street
  - All apartments in a large house share the house number
    - There is no mechanism to identify an apartment
  - A house can only be uniquely identified via “postcode – settlement name – street name – hous number”



Addresses in Germany



German postcode 55758

DETOUR

nany

https://www.suche-postleitzahl.org/

55758 hauptstraße 17

Wickenrodt 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Wickenrodt

Dickesbach 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Dickesbach

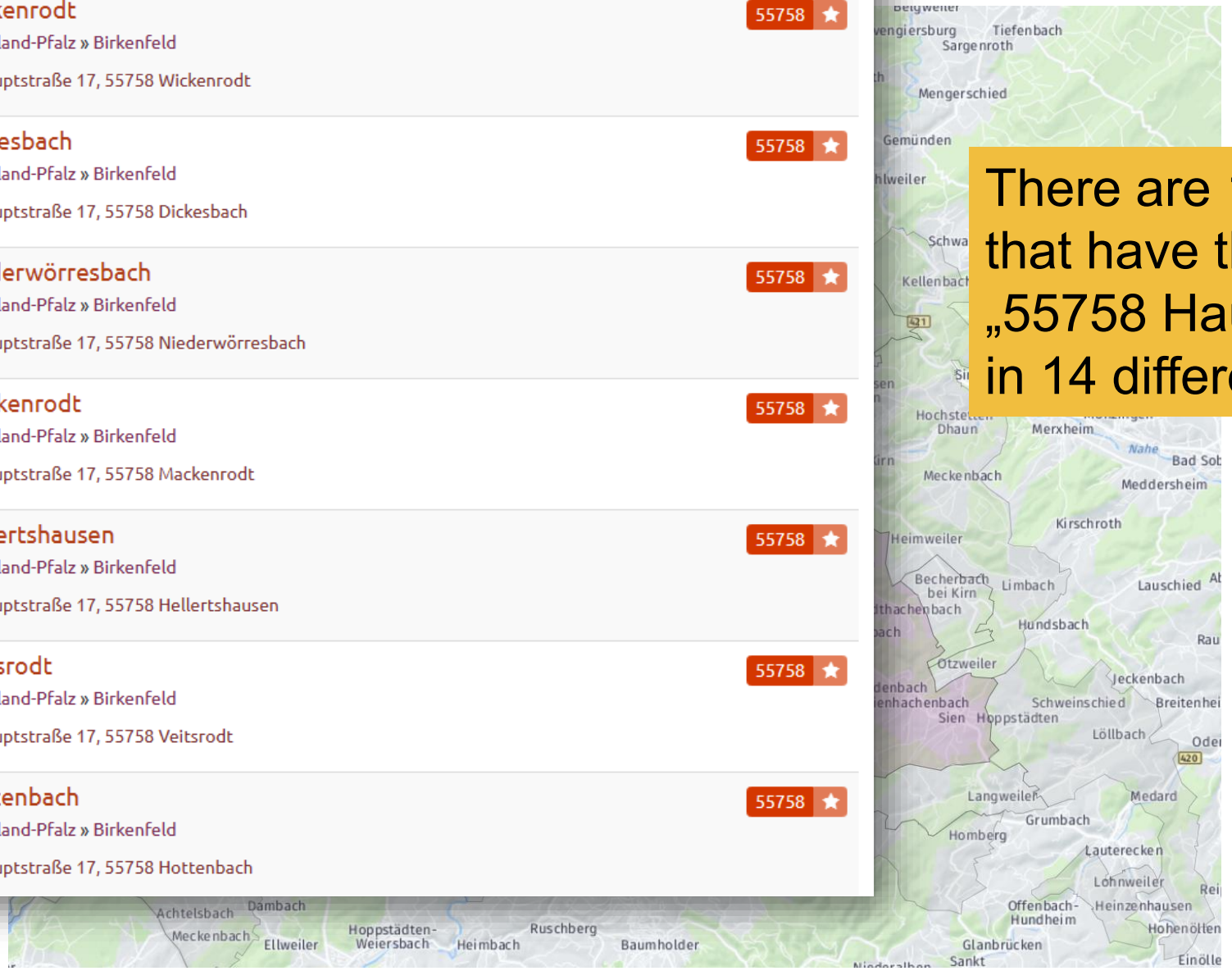
Niederwörresbach 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Niederwörresbach

Mackenrodt 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Mackenrodt

Hellertshausen 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Hellertshausen

Veitsrodt 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Veitsrodt

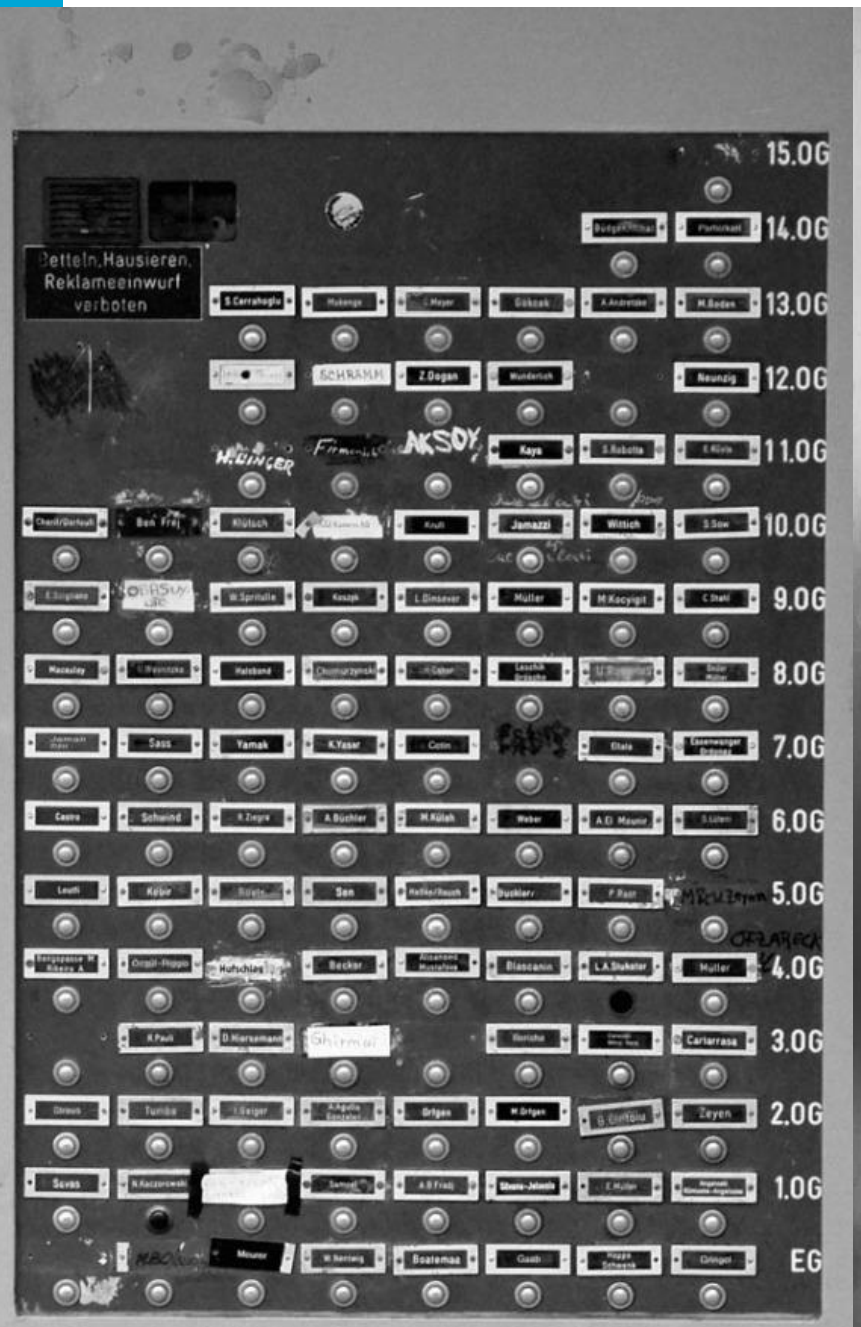
Hottenbach 55758
Rheinland-Pfalz » Birkenfeld
Hauptstraße 17, 55758 Hottenbach



There are 14 houses that have the address „55758 Hauptstraße 17“ in 14 different villages…

TU Delft Web Information Systems

DETOUR



Finding the house does NOT necessarily help you though…

Addresses in Japan

- National Institute for Informatics

  2 Chome-1-2 Hitotsubashi
  Chiyoda

  Tokyo 100-0003
  Japan

Have fun:
https://www.sljfaq.org/afaq/addresses.html

DETOUR

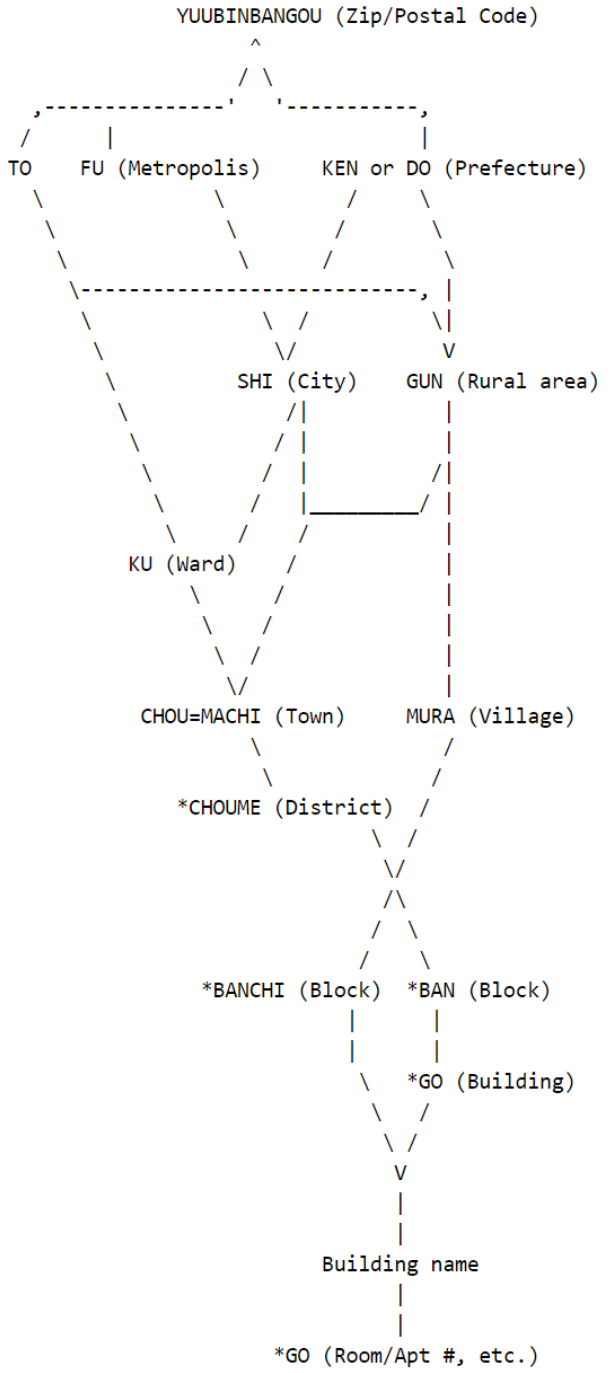
Addresses in Japa

- National Institute for Informatics

  2 Chome-1-2 Hitotsubashi
  Chiyoda

  Tokyo 100-0003
  Japan

Have fun:
https://www.sljfaq.org/afaq/addresses.html



TU Delft WIS Web Information Systems

DETOUR

Ad

- N

2

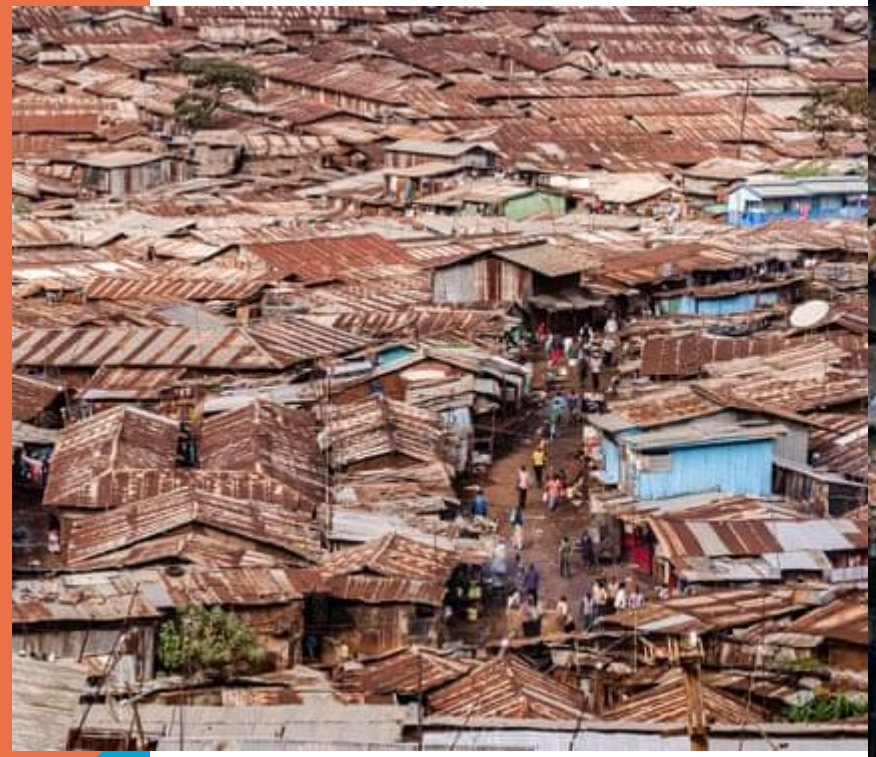
C

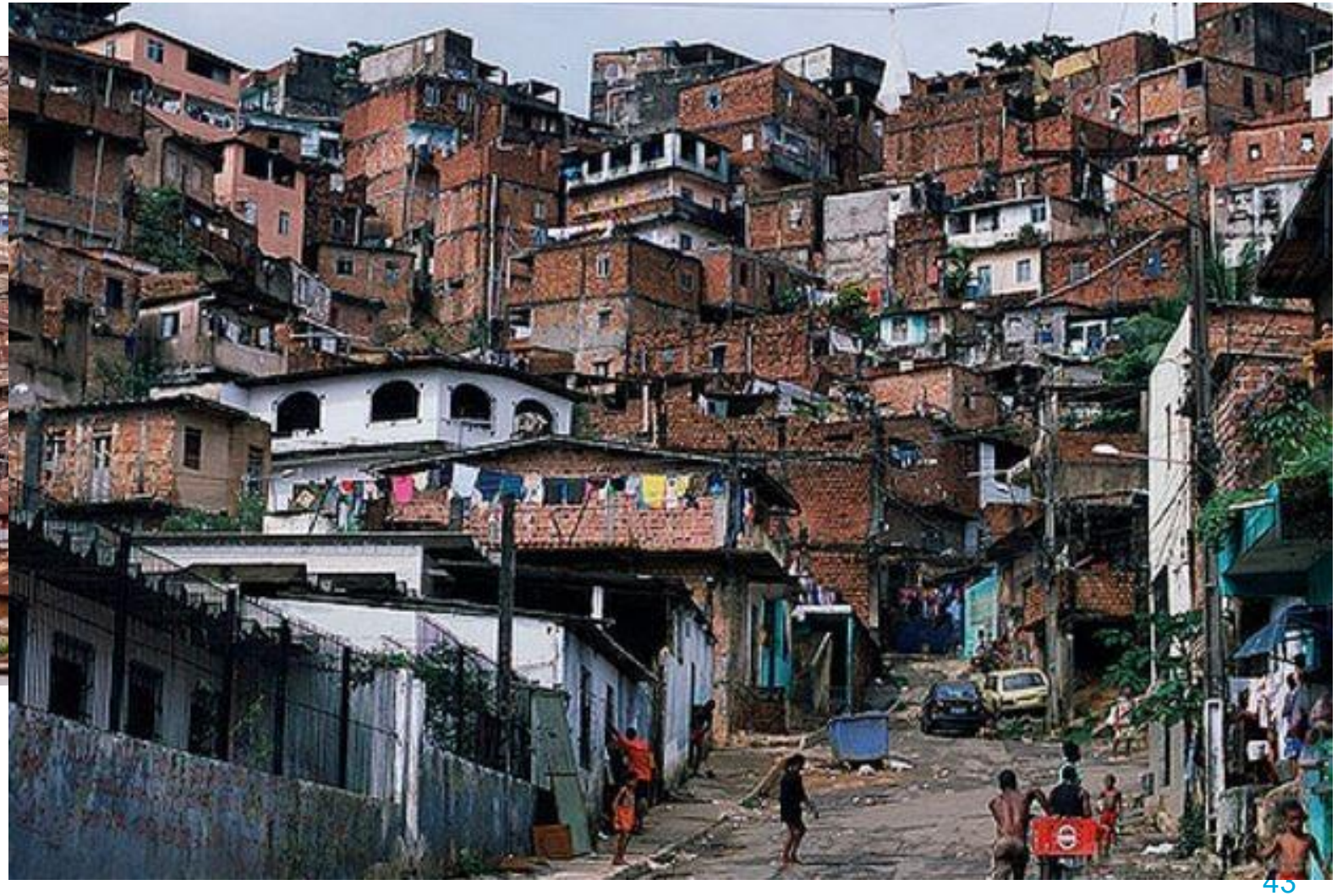
T

J

TU Delft Web Information Systems

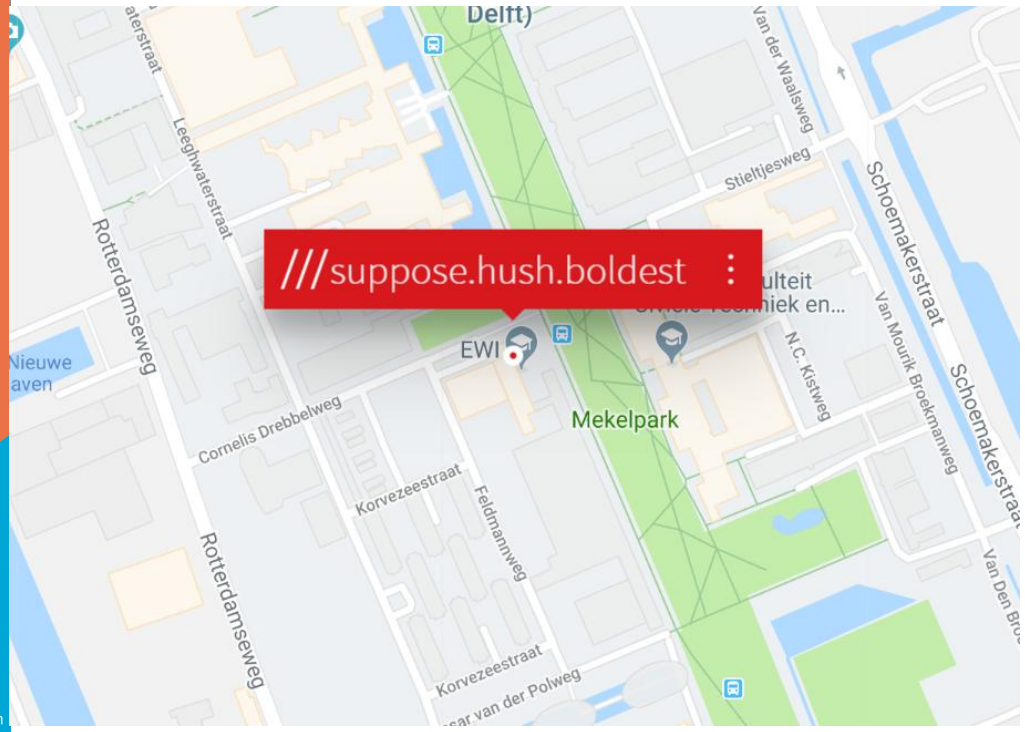
What about this????

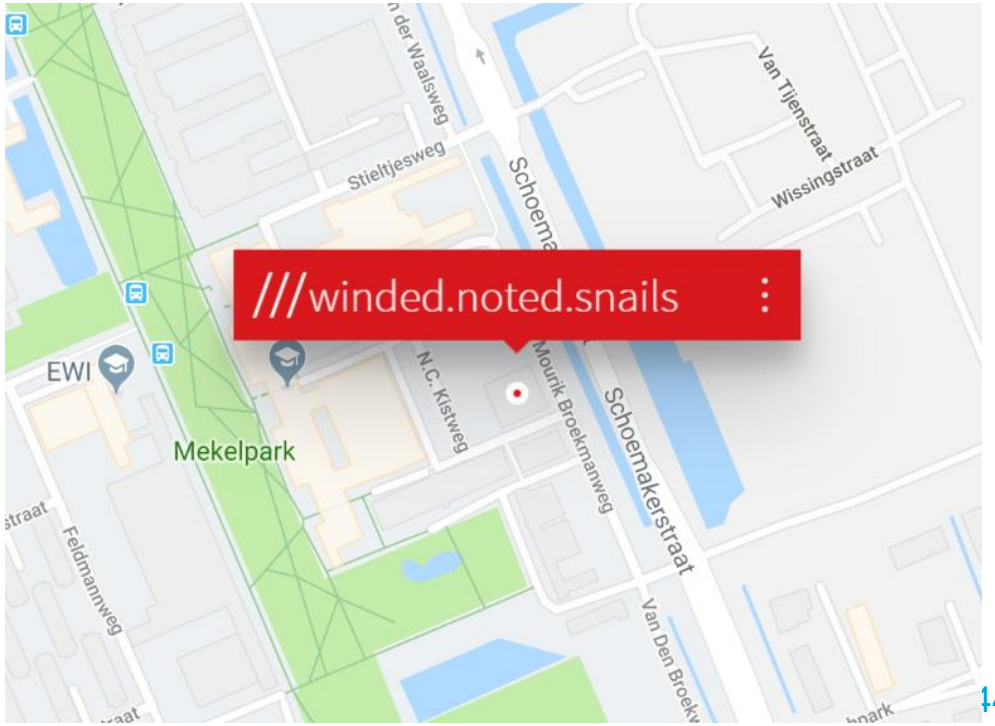




Global Addresses

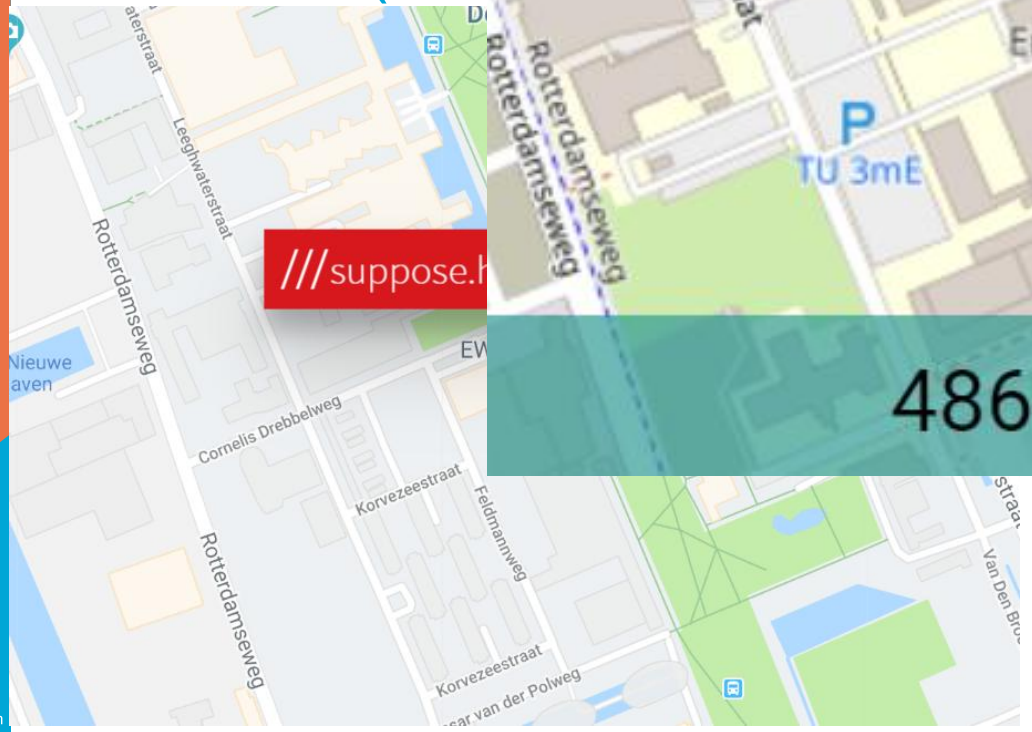
- What3Word.com
  - Functional Dependency between 3 English words, and any global 3x3m cell on the globe
    - (Word1, Word2, Word3) → (X-Cell, Y-Cell)

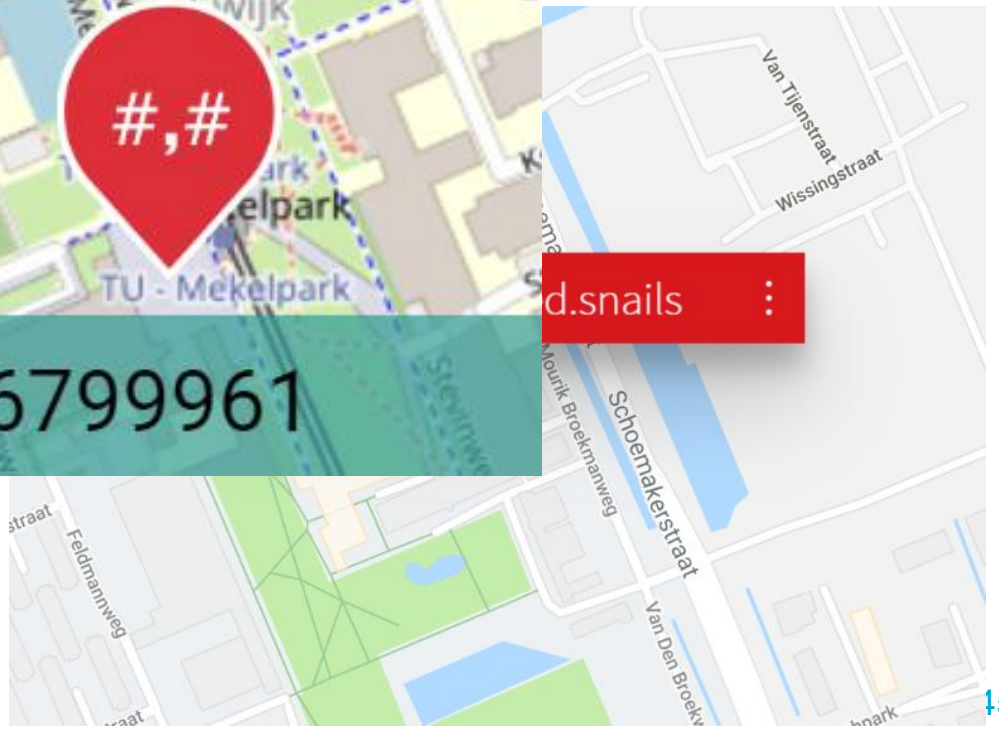




Global Addresses

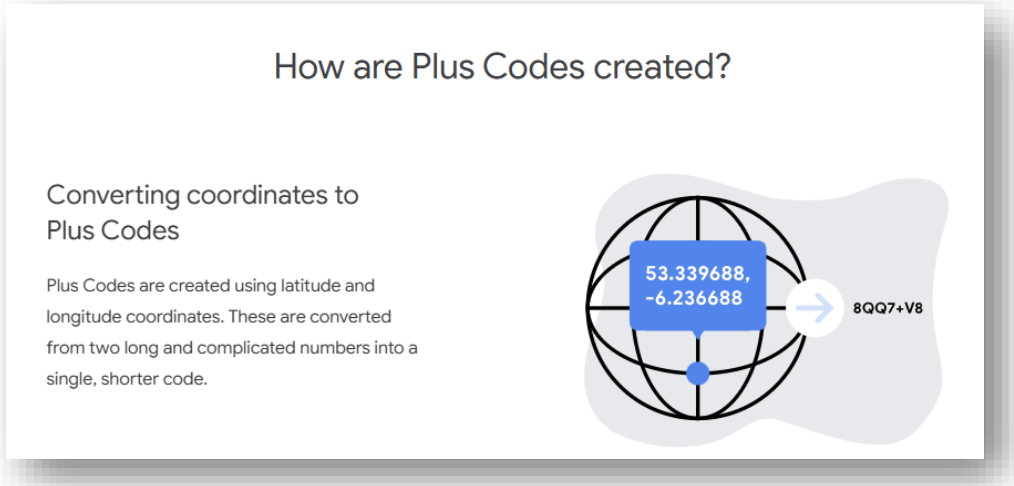
- What3
  - Functi ish words, and a
    - (Wo





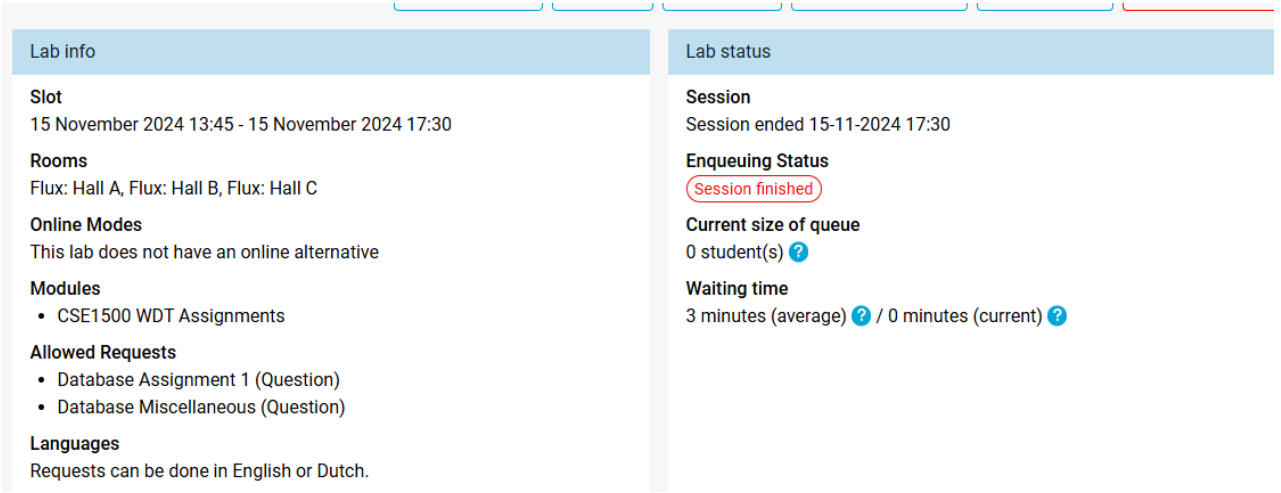
Global Adresses

- Google PlusCode
  - https://maps.google.com/pluscodes/



Queue

- The Queue System:
  - “There are courses, courses have sessions, and Students Visit TAs or TAs Visit Students”
  - Is this what we need or want?



IZ TU BS

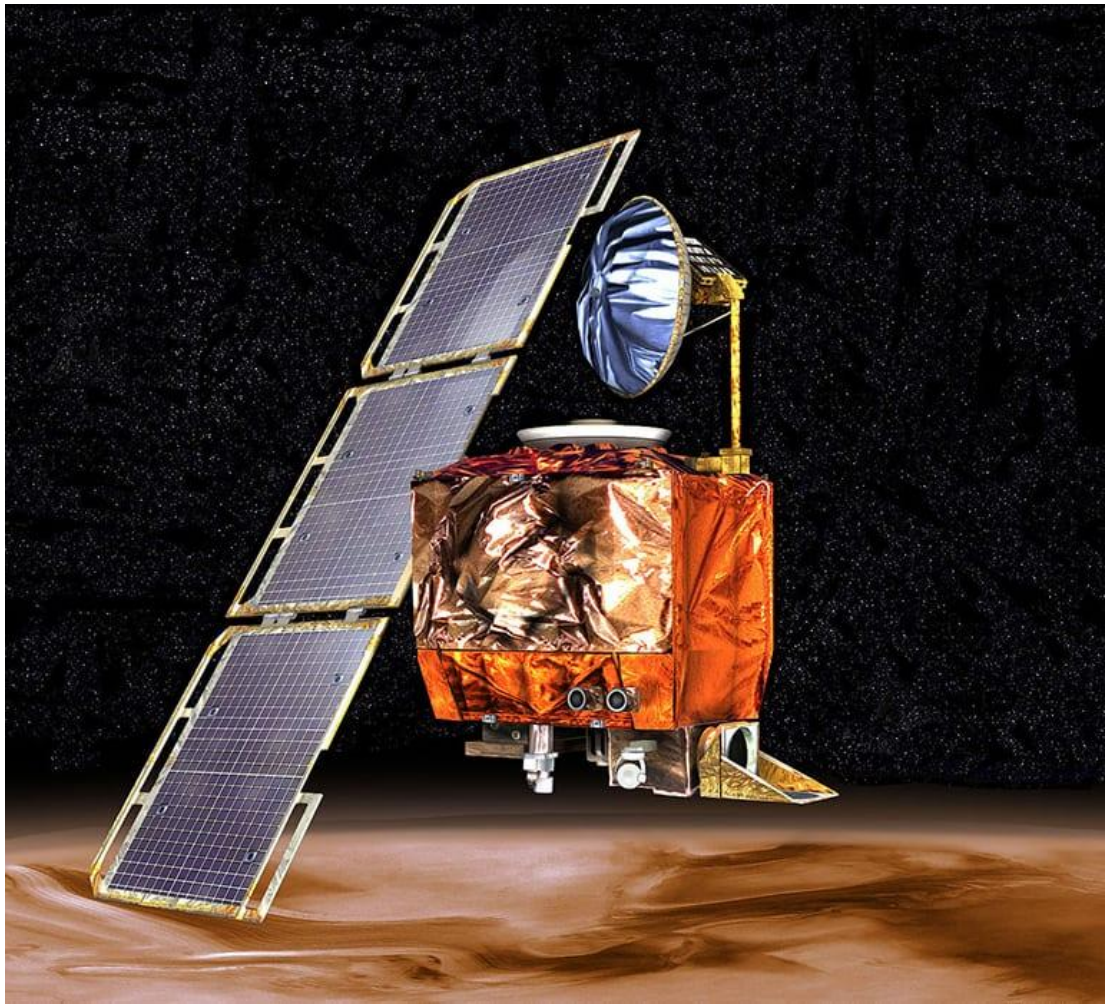
- The old version of the Informatikzentrum TU Braunschweig

Ariane 5 Acceleration: 1996

- Ariane 5 reused software from Ariane 4
  - Ariane flight V88 – Wikipedia
  - Ariane 5: „Vertical Velocity is a 64 Bit Float“
  - Ariane 4: „Vertical Velocity is a 16 Bit Integer“
  - Sadly, Ariane 5 goes a bit faster than the A4…

Mars Climate Orbiter 1999

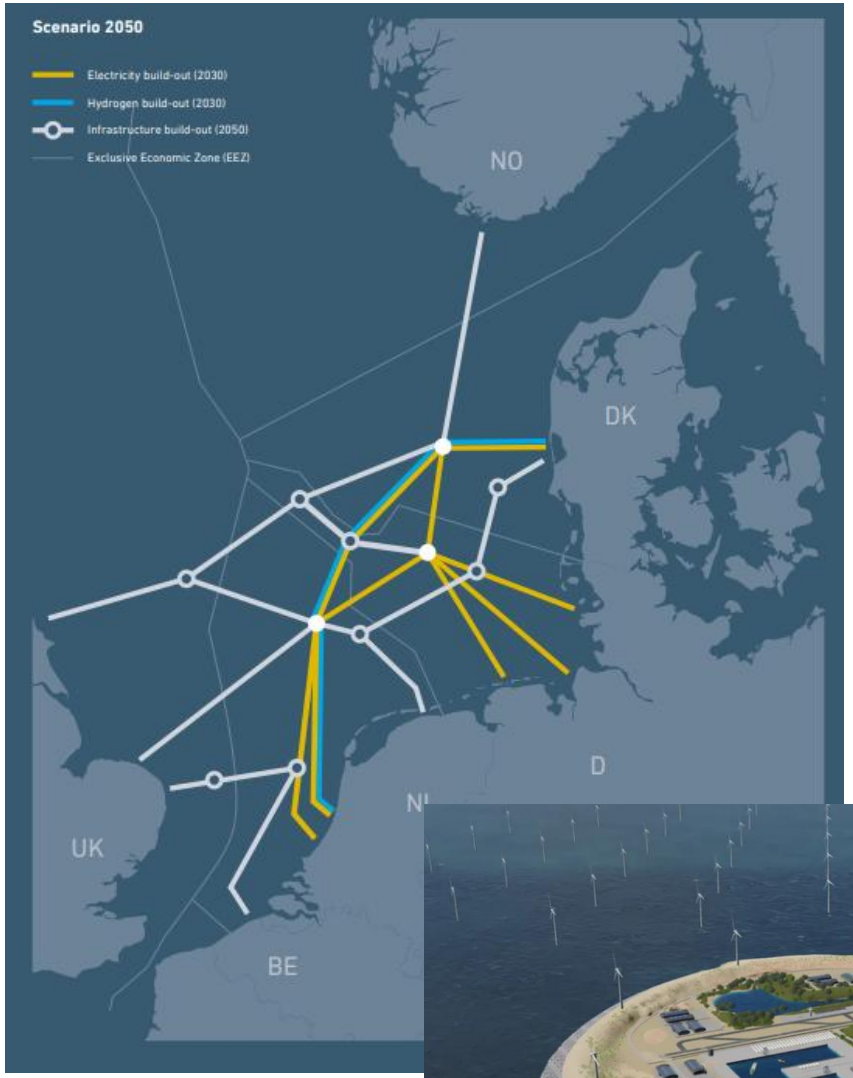
- "Fun" with the "Distance" attribute of the data schema ☺
  - "Please slow down when close to surface…"
  - Mars Climate Orbiter – Wikipedia
  - How NASA Lost Its Mars Climate Orbiter From a Metric Error

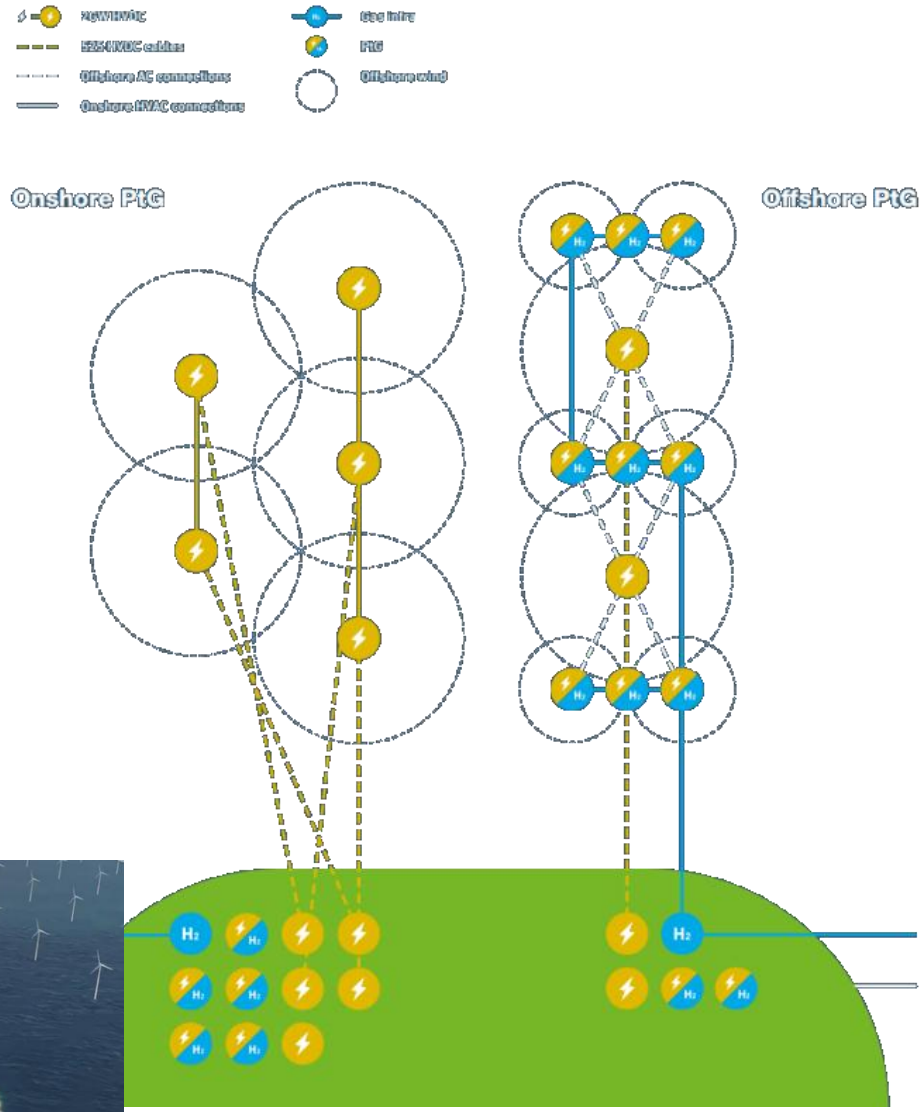


Classroom Discussion

See Brightspace “Extra Modelling Tasks.pdf”

Modelling Scenario





Modelling Scenario

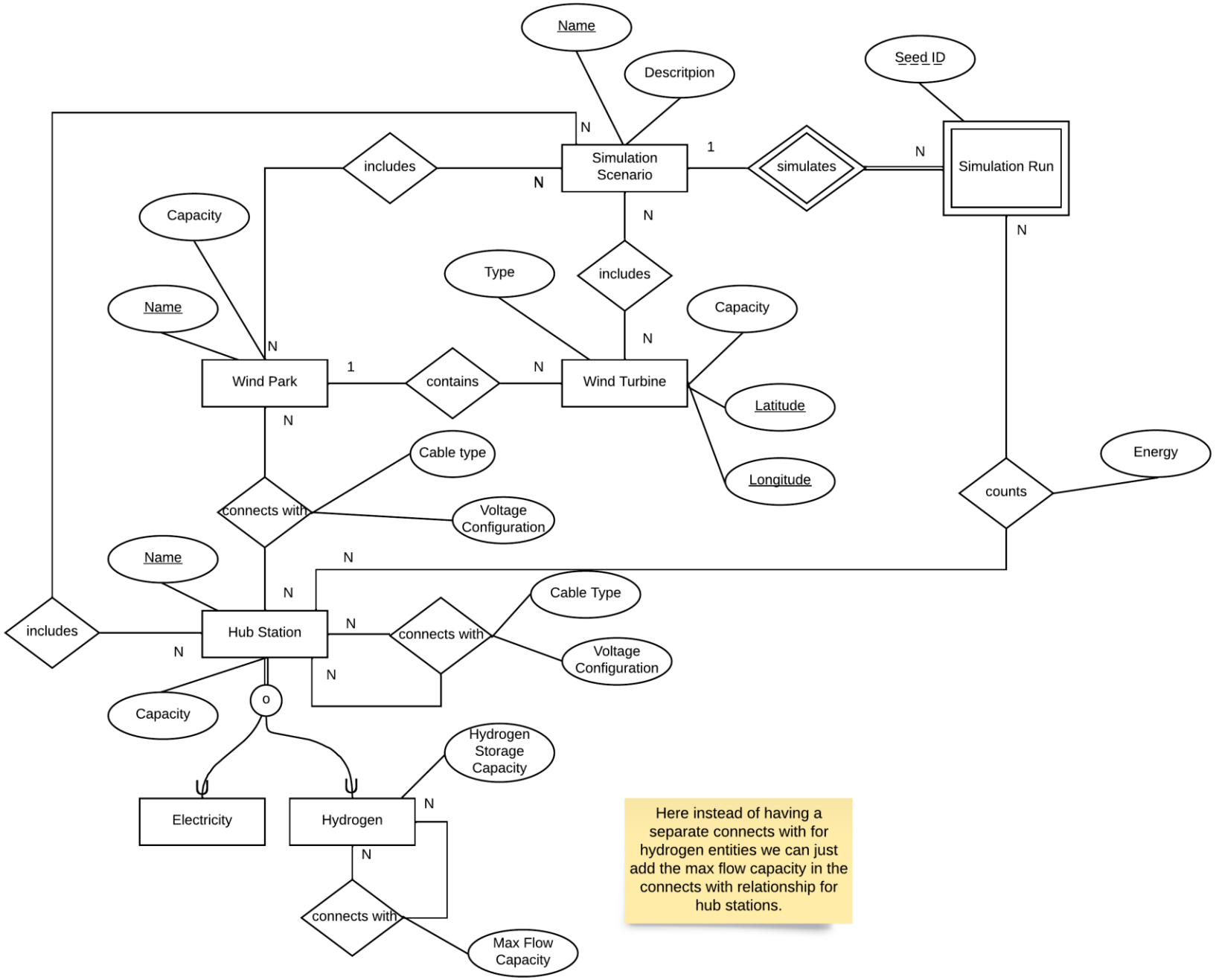
- The European North Sea Wind Power Hub aims at building a large-scale wind power infrastructure in the middle of the North Sea. The key idea is to have several large offshore wind parks, which are connected via a mesh of hub stations to the surrounding European countries.
  TU Delft assists with simulating possible configurations for wind parks, hubs, and connections to find the best layout and construction.
- Task: Create a **Conceptual EER** schema for supporting these simulations that satisfies the following requirements. Try to be as complete and efficient as possible. **Provide all primary keys and cardinalities! Comment your most important design choices and explain which described data requirements or constraints could not be represented in your diagram (if any).**

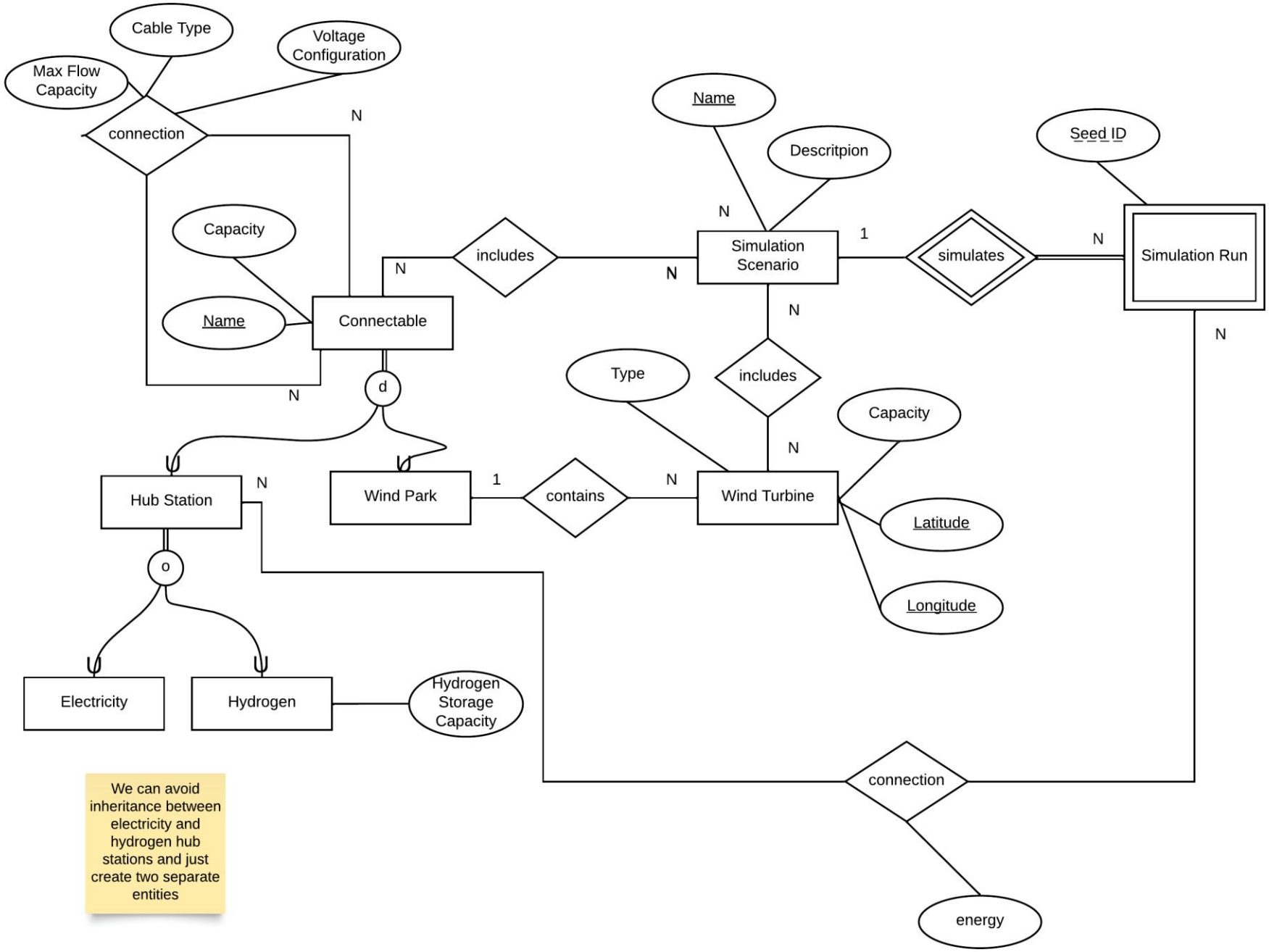
Modelling Scenario

In the system, different simulation scenarios can be defined, each with a unique name and an additional description. Each simulation scenario contains structures such as wind turbines, wind parks, and hub stations. For wind turbines, we store type, capacity, and their latitude and longitude. Latitude and longitude are unique and identify a wind turbine. Several wind turbines can be grouped into a wind park (each wind turbine must belong to exactly one wind park). Wind parks have a (unique) name, capacity.

Wind parks can be connected to one or more hub stations. These hub stations are responsible for transforming and distributing electric power. Hub stations can also be connected to other hub stations, and thus can form a complex network which allows for load balancing and risk sharing. Hub stations also have a unique name. We also store their capacity. Connections between hubs and/or wind parks can use different types cables and voltage configurations: AC (alternating current) cables or DC (direct current) cables, both in either low voltage and high voltage configuration. There can also be hub stations which additionally to transforming electricity also produce hydrogen (H2) from electricity and seawater. Such hydrogen stations can be connected to other hydrogen stations via special hydrogen pipeline connections, each having a maximal flow capacity. Hydrogen stations also have a hydrogen storage capacity noted in the database.

For each simulation scenario (as described above), several simulation runs can be executed on the Delft supercomputer cluster. Each simulation run simulates the (artificial) weather over the course of one week. The weather used for that simulation is given by a seed id (i.e., with that seed id, all weather for that week can deterministically be recreated). The seed id and the simulation scenario identify each run. Furthermore, for each simulation run, we store the overall energy (and/or H2) produced or collected at each individual hub station. This data helps to assess the resilience and performance of the planned infrastructure.


Name
Descritpion
Seed ID
Simulation Scenario
includes
simulates
Simulation Run
Capacity
Type
includes
Name
Capacity
Wind Park
contains
Wind Turbine
Latitude
Longitude
Cable type
Energy
counts
connects with
Voltage Configuration
Name
Cable Type
includes
Hub Station
connects with
Voltage Configuration
Capacity
Hydrogen Storage Capacity
Electricity
Hydrogen
connects with
Max Flow Capacity
Here instead of having a separate connects with for hydrogen entities we can just add the max flow capacity in the connects with relationship for hub stations.


Cable Type
Voltage Configuration
Max Flow Capacity
connection
N
Name
Descritpion
Seed ID
N
1
N
Capacity
includes
Simulation Scenario
simulates
Simulation Run
N
N
N
Name
Connectable
N
N
d
Type
includes
Capacity
N
Hub Station
N
Wind Park
1
contains
N
Wind Turbine
Latitude
Longitude
o
Electricity
Hydrogen
Hydrogen Storage Capacity
We can avoid inheritance between electricity and hydrogen hub stations and just create two separate entities
connection
energy

Modelling Everything

TU Delft WIS Web Information Systems

Ontologies

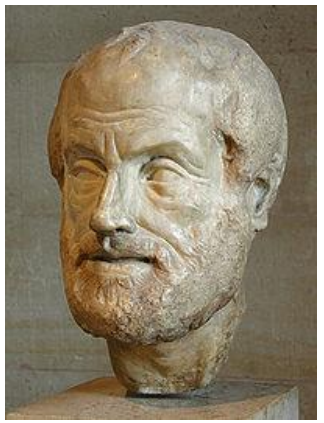
- In computer science ontologies are formal specifications of a shared conceptualization
  - Basically, an ontology provides a shared vocabulary and descriptions of the real world
    - It can be used to define the type of objects and/or concepts that exist, and their properties and relations
    - More “expressive” than schemas
    - Ontologies are often equated with taxonomic hierarchies
      - But this definition is far too narrow – ontologies are so much more…
      - See ontologies in philosophy
  - Domain ontologies model the real world with respect to a specific domain
    - This also disambiguates most terms
    - But domain ontologies are not compatible with each other…

History of Ontologies

- Science and philosophy always strived to **explain** the world and the nature of being
  - First formal school of studies: **Aristotle's metaphysics** ("beyond the physical", ca. 360 BC)
  - Traditional branches of metaphysics:
    - **Ontology** (λόγος: word, science and ὄντος: of being)
      - Study of being and existence
    - Natural theology
      - Study of God, nature and creation
    - Universal science
      - "First Principles", logics

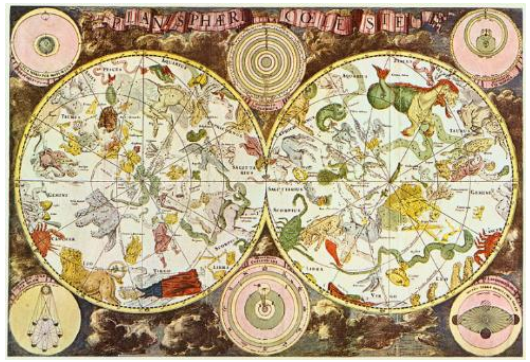
History of Ontologies

- **Ontology** tries to describe everything which **is** (exists), and its relation and categorization with respect to other things in existence
  - What is **existence**? Which **things** exists? Which are **entities**?
  - Which **entities** are fundamental?
  - What is a **physical object**?
  - How do the **properties** of an object relate to the object itself? What features are the **essence**?
  - What does it mean when a physical object exists?
  - What constitutes the **identity** of an object?
  - When does an object **go out of existence**, as opposed to merely **change**?



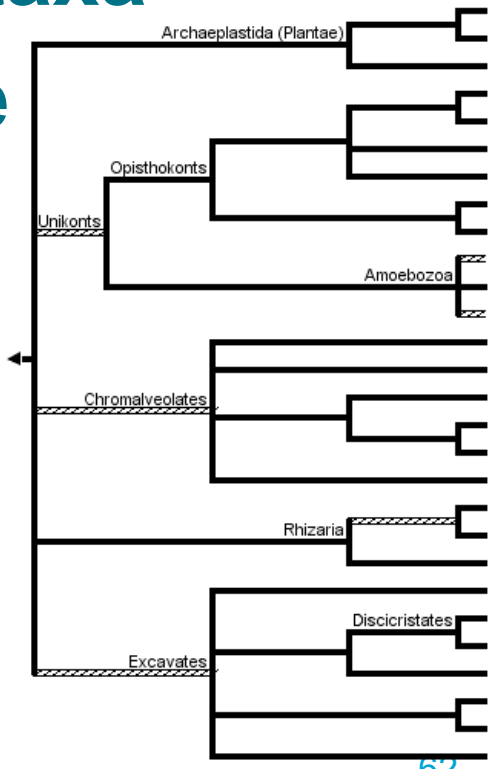
History of Ontologies

- Parts of metaphysics evolved into **natural philosophy**
  - Study of **nature** and the **physical universe**
  - In the late 18th century, it became just '**science**'
  - Ontology is still a dominant concept in science
    - Representation of all knowledge about things



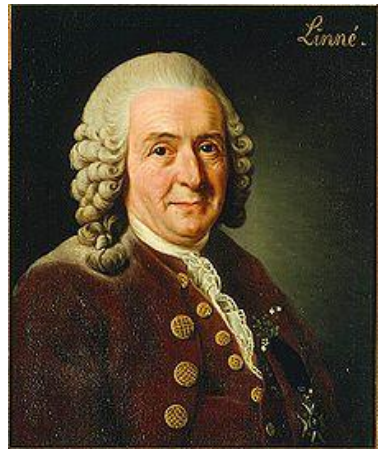
History of Ontologies

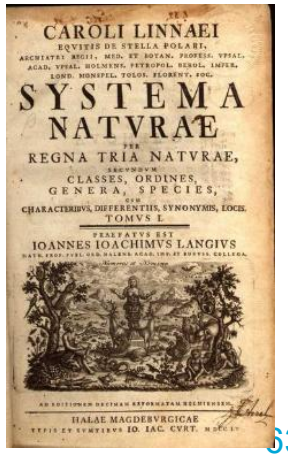
- **Taxonomies** (τάξις : arrangement, *νόμος*: law) are part of ontology
  - Groups things with similar properties into **taxa**
  - Taxa are put into a **hierarchical structure**
    - Hierarchy represents **supertype-subtype** relationships
    - Represents a **specialization** of taxa, starting with the most general one



History of Ontologies

- Example: **Linnaean Taxonomy**
  - Classification of all living things by Carl von Linné in 1738
  - Classification into multiple hierarchy layers
    - Domain, Kingdom, Phylum, Subphylum, Class, Cohort, Order, Suborder, Infraorder, Superfamily, Family, Genus, Species
  - Each layer adds additional properties and restrictions





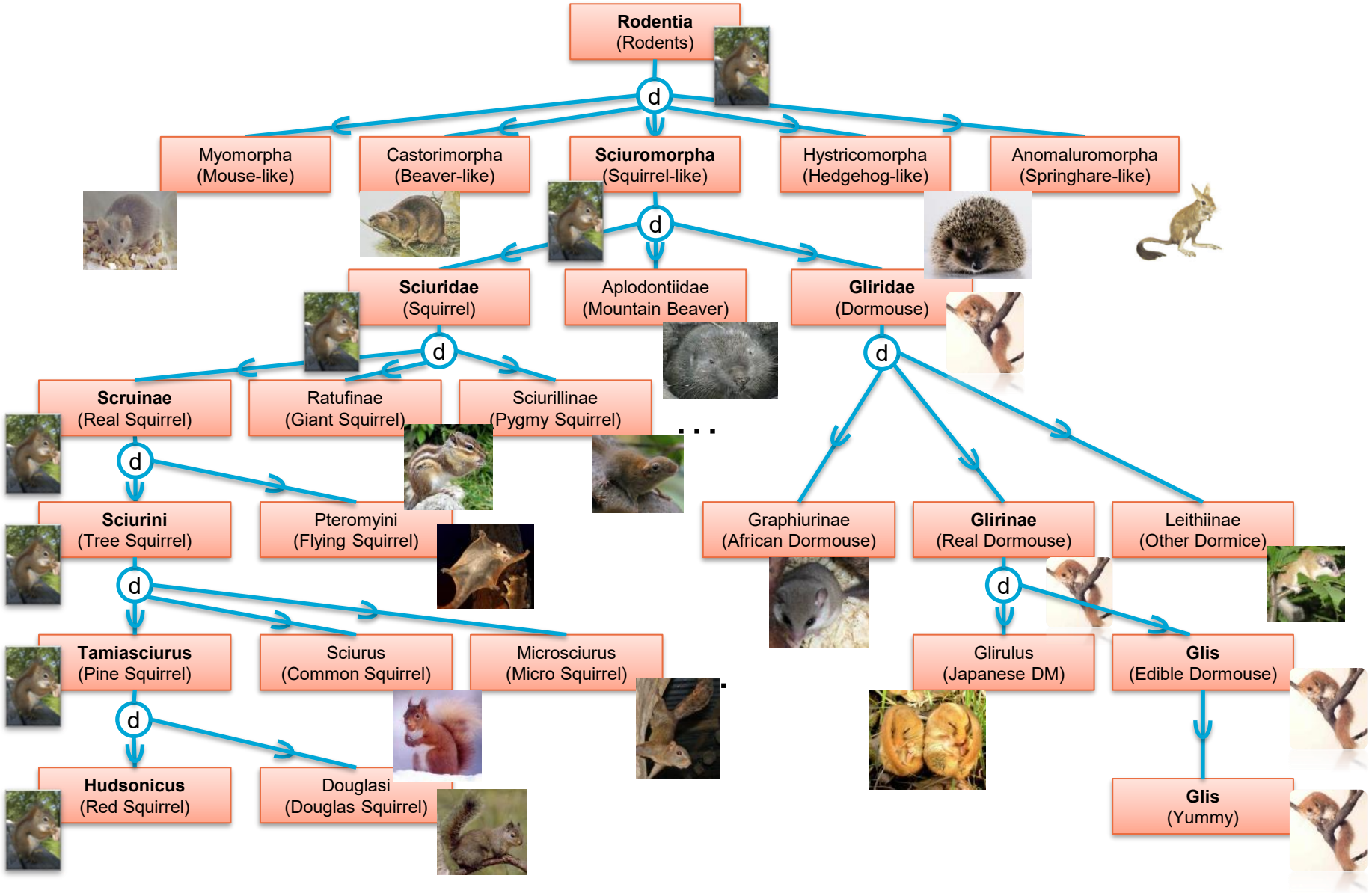
History of Ontologies

- Example: **Red Squirrel**
  (Binomial Name: Tamiasciurus hudsonicus)
  - Kingdom: **Animals**
  - Phylum: **Chordata** (with ***backbone***)
  - Class: **Mammalia** (with backbone, ***nursing its young***)
  - Order: **Rodentia** (backbone, nursing its young, ***sharp front teeth***)
  - Suborder: **Scriuomorpha** (backbone, nursing its young, sharp front teeth, ***like squirrel***)
  - Family: **Scriudae** (backbone, nursing its young, sharp front teeth, like squirrel, ***bushy tail & lives on trees (i.e., real squirrel)***)
  - Genus: **Tamiasciurus** (backbone, nursing its young, sharp front teeth, like squirrel, bushy tail & trees, ***from N-America***)
  - Species: **Hudsonicus** (backbone, nursing its young, sharp front teeth, like squirrel, bushy tail & trees, from N-America, ***brown fur with white belly***)

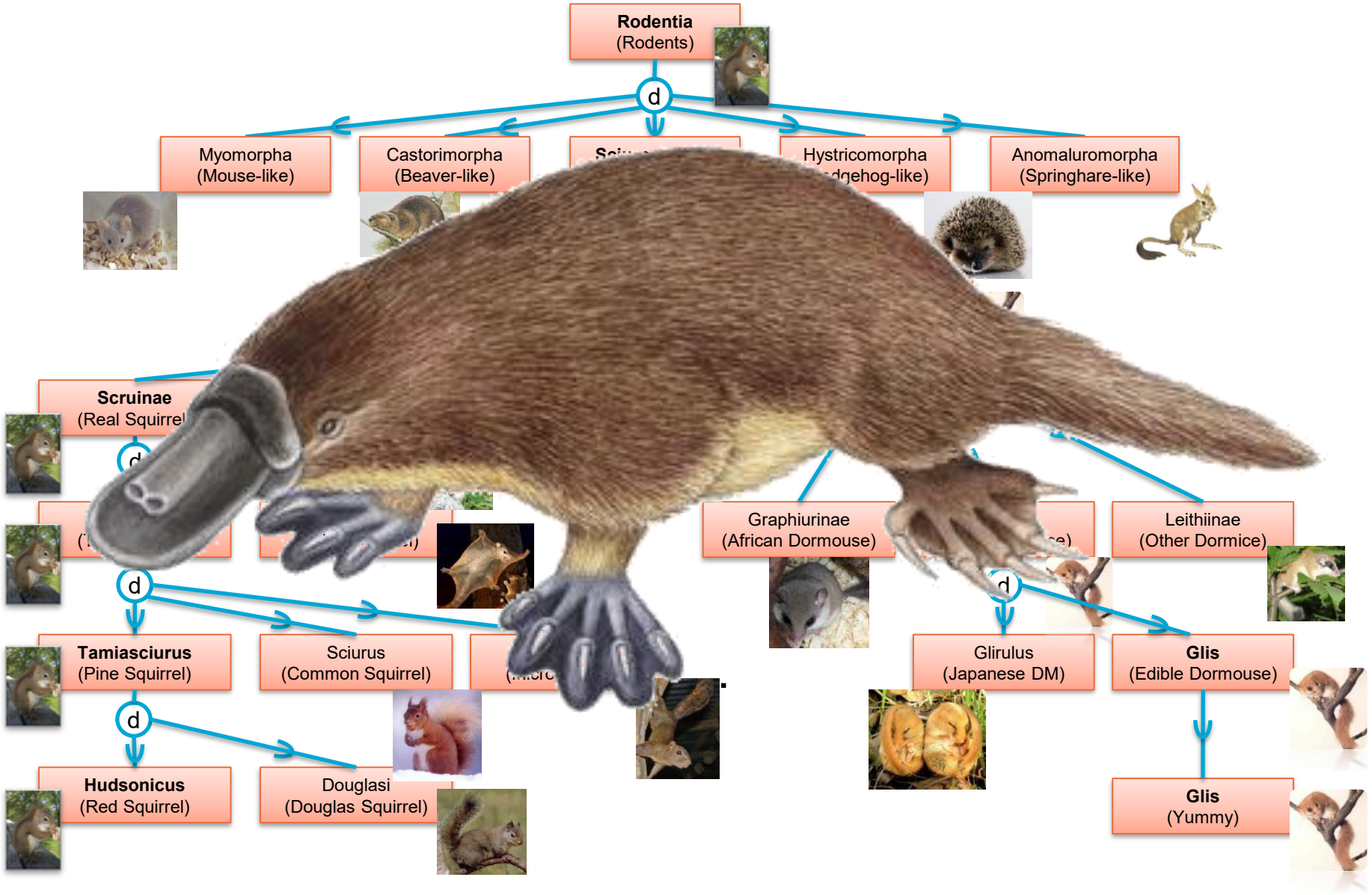
History of Ontologies

- Example: **Edible Dormouse** *(Binomial Name: Glis Glis)*
  - Kingdom**: Animals**
  - Phylum**: Chordata** (with ***backbone***)
  - Class: **Mammalia** (with backbone, ***nursing its young***)
  - Order: **Rodentia** (backbone, nursing its young, ***sharp front teeth***)
  - Suborder: **Scriuomorpha** (backbone, nursing its young, sharp front teeth, ***like squirrel***)
  - Family: **Gliradae** (backbone, nursing its young, sharp front teeth, like squirrel, ***sleeps long***)
  - Genus: **Glis** (backbone, nursing its young, sharp front teeth, bushy tail, like squirrel, ***eaten by Romans***)
  - Species: **Glis** (backbone, nursing its young, sharp front teeth, bushy tail, climbs trees, ***nothing more to classify***)

History of Ontologies

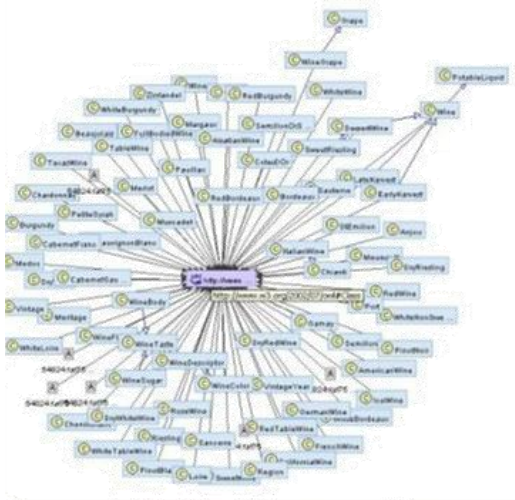


History of Ontologies



Ontologies in CS

- Recently, creating **ontological models** became fashionable in CS
  - So called **ontologies**
  - Widely used in e.g., medical informatics, bio-informatics, Semantic Web
- In addition to *normal* data models, ontologies offer **reasoning capabilities**
  - Allow to classify instances automatically
  - Allow to extract additional facts from the model
- In CS, ontologies are usually modeled using **special languages**
  - e.g., OWL, DAML+OIL, IDEF



THIS IS HOME STUDY MATERIAL
Also covered in TUTORIALS

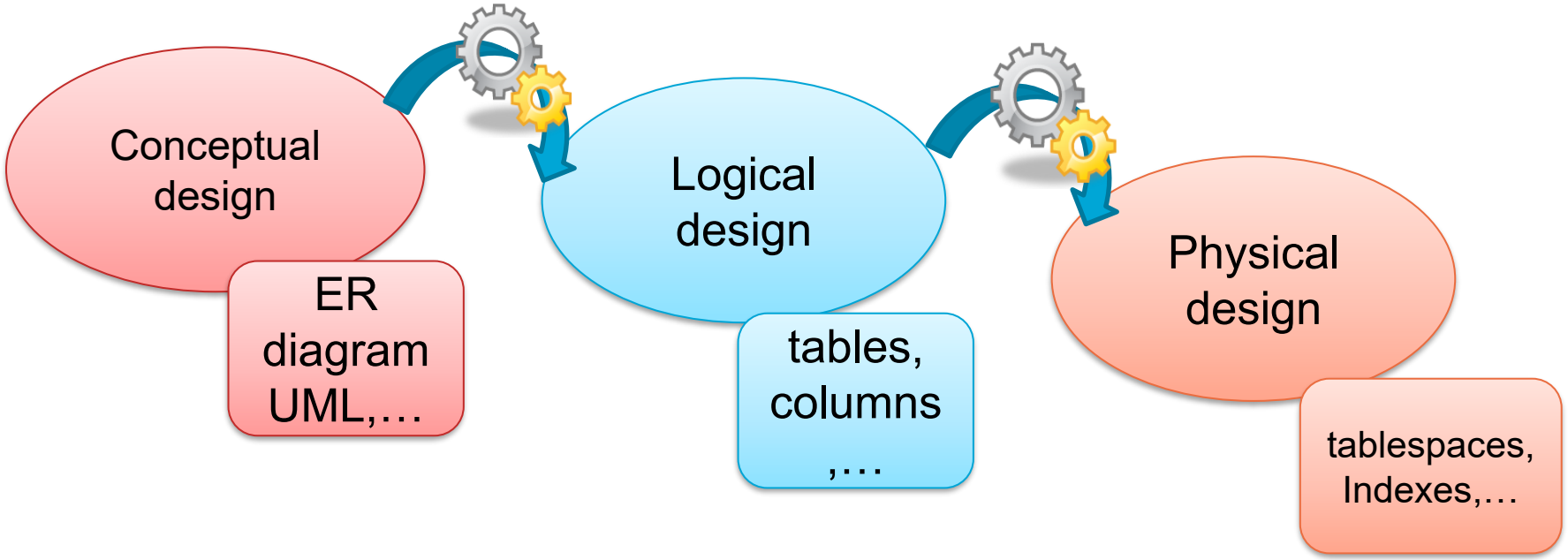
See WDT_LogicalModelling-20231122_101314-Meeting Recording.mp4 on Brightspace

Conceptual to Logical Schemas

TU Delft WIS Web Information Systems

Conversion from ER

- After modeling a conceptual schema (e.g., using an **ER diagram**), the schema can be (semi-) **automatically** transformed into a **relational schema**



AGAIN

- Conceptual Schema
  - ”Which entities should be stored, what are their properties, and how are they related?”
- (Relational) Logical Schema
  - “Which relations should exist, which attributes do they have, what are the domains (data types) of the attributes, and constraints should hold?”
- Physical Schema
  - “Where and how to store relations, what to index, what meta-data / statistics to collect, etc.?”

Conceptual To Logical

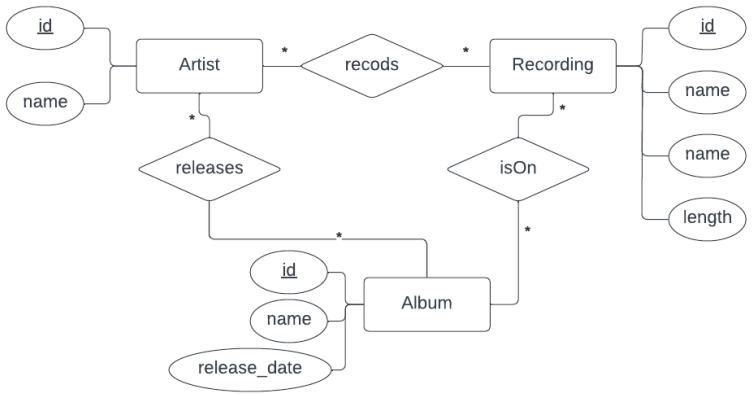
- Question: How to convert a conceptual ER schema to relational logical schema?
  - We can automatically convert a conceptual ER model to relations
    - Some heuristics follow…
    - However, quite often the result will not be as desired
    - Therefore, still some manual optimization and steering is beneficial
    - While designing a model, it might be very beneficial to keep the result relations and the desired queries in mind…

The Relational Models Stores Data in Tables

- We do this in more detail later…
- **Tables** (or also called Relations) have a name and several columns and constraints
  - Columns (or also called Attributes)
    - Attributes have a domain (type), e.g., characters, numbers, etc.
    - Attributes are the columns of the table
  - Additional Column Constraints
    - Primary Key Constraints (PK)
      - A column or combination of columns that contain values that uniquely identify each row in the table
      - Each table should have one Primary Key Column or PK Column Set
    - Foreign Key Constraints (FK)
      - Foreign Keys “link” tables
      - A foreign key (FK) is a column or combination of columns that is used to establish and enforce a link between the data in two tables to control the data that can be stored in the foreign key table.
      - Used to enforce referential integrity
    - Additional Columns Constraints like unique, not null, etc.

Example: A logical schema of our music database using the relational model

Conceptual Schema



Focus on Entity Types and their relationships

Logical (Relational) Schema

Focus the resulting relations / tables and constraints

Artist

PK	id
	name

l_artist_recording

PK,FK	a_id
PK,FK	r_id

Recording

PK	id
	name
	rating
	length

l_artist_album

PK,FK	a_id
PK,FK	alb_id

Album

PK	id
	name
	release_date

l_album_recording

PK,FK	alb_id
PK,FK	r_id
	position

Artist

id	name
85	Madness
86	Qaballah Steppers
87	Steve Rossiter
88	Enya
89	Madonna
91	Bill Cosby
92	Yanni
93	Metallica
94	John Williams
96	The Bangles
98	Burl Ives
99	Bing Crosby
101	Frank Sinatra
104	[Disney]
110	Harry Connick, Jr.
111	Mariah Carey
116	Enigma
119	George Winston
121	The Chieftains
122	New Edition
123	Cliff Richard
125	Dan Fogelberg
127	James Taylor
129	Simon & Garfunkel

l_artist_recording

a.id	r.id
94	257208
94	334474
94	447670
94	453112
94	453118
94	453119
94	464602
94	464603
94	464604
94	496934
94	515742
94	515746
94	515748
94	532750
94	532751
94	532752
94	532754
94	532755
94	532757

Recording

record_id	record_name	rating...	length
257208	Superman: Main Theme	4	04:26
334474	Prologue	6	02:12
447670	Duel of the Fates	7	04:14
453112	Opening Titles	8	00:33
453118	Welcome to Jurassic Park	8	07:54
453119	My Friend, the Brachiosaurus	6	04:16
464602	Ah, Rats!!!	8	03:41
464603	Escape From Venice	6	04:24
464604	No Ticket	6	02:46
496934	E.T.'s New Home	4	01:38
515742	Yoda's Theme	6	03:29
515746	Yoda and the Force	6	04:02
515748	Lando's Palace	6	03:53
532750	Star Wars Main Title / Ambush on Corus…	8	03:46
532751	Across the Stars (Love Theme From Atta…	8	05:33
532752	Zam the Assassin / The Chase Through C…	10	11:07
532754	Departing Coruscant	6	01:44
532755	Anakin and Padmé	6	03:56
532757	The Meadow Picnic	8	04:14
532758	Bounty Hunter's Pursuit	8	03:23
532759	Return to Tatooine	8	06:56
532760	The Tusken Camp / The Homestead	6	05:54
532761	Love Pledge / The Arena	8	08:29
532762	Confrontation With Count Dooku / Finale	8	10:45

Artist

id	name
85	Madness
86	Qaballah Steppers
87	Steve Rossiter
88	Enya
89	Madonna
91	Bill Cosby
92	Yanni
93	Metallica
94	John Williams
96	The Bangles
98	Burl Ives
99	Bing Crosby
101	Frank Sinatra
104	[Disney]
110	Harry Connick, Jr.
111	Mariah Carey
116	Enigma
119	George Winston
121	The Chieftains
122	New Edition
123	Cliff Richard
125	Dan Fogelberg
127	James Taylor
129	Simon & Garfunkel

l_artist_recording

a.id	r.id
94	257208
94	334474
94	447670
94	453112
94	453118
94	453119
94	464602
94	464603
94	464604
94	496934
94	515742
94	515746
94	515748
94	532750
94	532751
94	532752
94	532754
94	532755
94	532757

Recording

record_id	record_name	rating...	length
257208	Superman: Main Theme	4	04:26
334474	Prologue	6	02:12
447670	Duel of the Fates	7	04:14
453112	Opening Titles	8	00:33
453118	Welcome to Jurassic Park	8	07:54
453119	My Friend, the Brachiosaurus	6	04:16
464602	Ah, Rats!!!	8	03:41
464603	Escape From Venice	6	04:24
464604	No Ticket	6	02:46
496934	E.T.'s New Home	4	01:38
515742	Yoda's Theme	6	03:29
515746	Yoda and the Force	6	04:02
515748	Lando's Palace	6	03:53
532750	Star Wars Main Title / Ambush on Corus…	8	03:46
532751	Across the Stars (Love Theme From Atta…	8	05:33
532752	Zam the Assassin / The Chase Through C…	10	11:07
532754	Departing Coruscant	6	01:44
532755	Anakin and Padmé	6	03:56
532757	The Meadow Picnic	8	04:14
532758	Bounty Hunter's Pursuit	8	03:23
532759	Return to Tatooine	8	06:56
532760	The Tusken Camp / The Homestead	6	05:54
532761	Love Pledge / The Arena	8	08:29
532762	Confrontation With Count Dooku / Finale	8	10:45

This table requires foreign key constraints:
e.g, there should be no a.id without a matching artist.id,
and no r.id without a matching recording.record_id!
This is called Referential Integrity

Artist

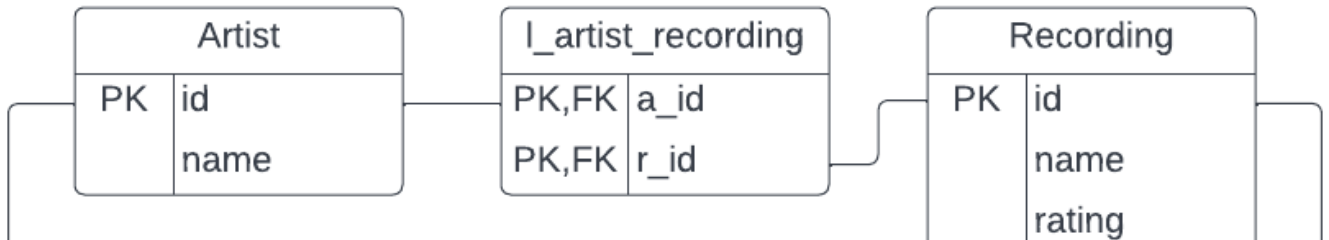
id	name
85	Madness
86	Qaballah Steppers
87	Steve Rossiter
88	Enya
89	Madonna
91	Bill Cosby
92	Yanni
93	Metallica
94	John Williams
96	The Bangles
98	Burl Ives
99	Bing Crosby
101	Frank Sinatra
104	[Disney]
110	Harry Connick, Jr.
111	Mariah Carey
116	Enigma
119	George Winston
121	The Chieftains
122	New Editi…
123	Cliff Ric…
125	Dan Fogel…
127	James Tay…
129	Simon & G…

l_artist_recording

a.id	r.id
94	257208
94	334474
94	447670
94	453112
94	453118
94	453119
94	464602
94	464603
94	464604
94	496934
94	515742
94	515746
94	515748
94	532750
94	532751
94	532752
94	532754

Recording

record_id	record_name	rating...	length
257208	Superman: Main Theme	4	04:26
334474	Prologue	6	02:12
447670	Duel of the Fates	7	04:14
453112	Opening Titles	8	00:33
453118	Welcome to Jurassic Park	8	07:54
453119	My Friend, the Brachiosaurus	6	04:16
464602	Ah, Rats!!!	8	03:41
464603	Escape From Venice	6	04:24
464604	No Ticket	6	02:46
496934	E.T.'s New Home	4	01:38
515742	Yoda's Theme	6	03:29
515746	Yoda and the Force	6	04:02
515748	Lando's Palace	6	03:53
532750	Star Wars Main Title / Ambush on Corus…	8	03:46
532751	Across the Stars (Love Theme From Atta…	8	05:33
532752	Zam the Assassin / The Chase Through C…	10	11:07
532754	Departing Coruscant	6	01:44
532755	Anakin and Padmé	6	03:56
532757	The Meadow Picnic	8	04:14
532758	Bounty Hunter's Pursuit	8	03:23
…	…ne	8	06:56
…	/ The Homestead	6	05:54
…	e Arena	8	08:29
…	th Count Dooku / Finale	8	10:45

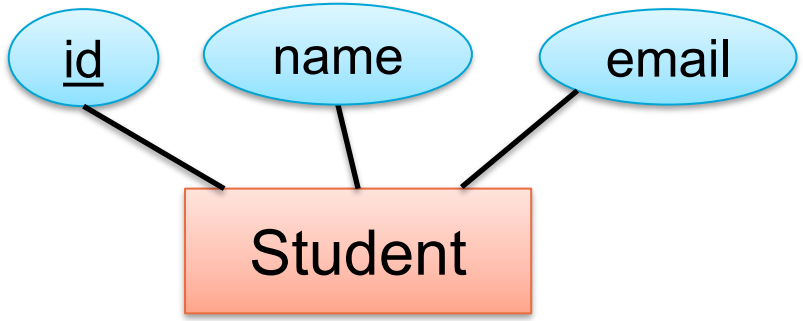


Conceptual To Logical

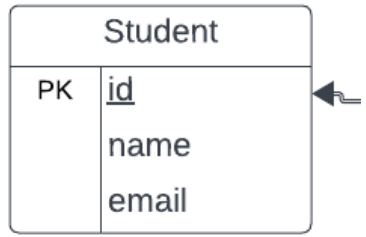
- Converting a simple Entity Type into a relation schema:

It's as simple as that

Conceptual Schema
(EER Diagram)



Logical Relational Schema
(ER Diagram)



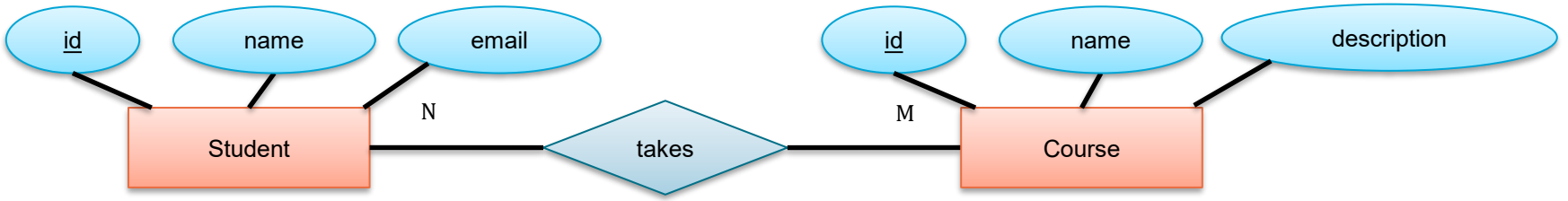
Student(id, name, email)

Logical Relational Schema
(Text Representation)

Conceptual To Logical

- Converting an N:M relationship type into a logical relation schema:
  - Relationship type becomes a separate table
    - Foreign key constraints ensure that the two entity types are properly linked
    - The new relationship table uses a composite primary key

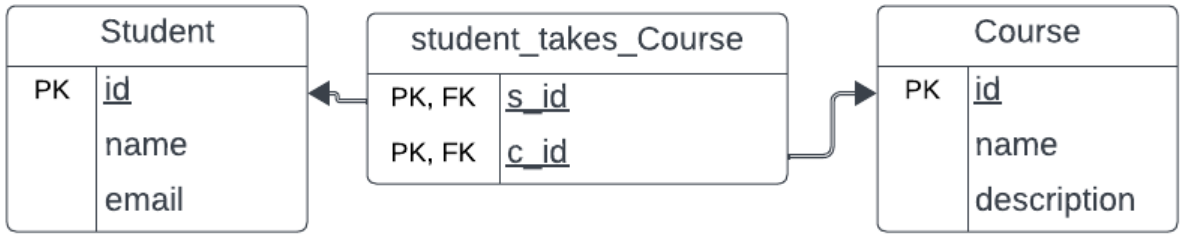
Conceptual Schema
(EER Diagram)



Student(id, name, email)
student_takes_course(s_id → Student(id), c_id→ Course(id))
Course(id, name, description)

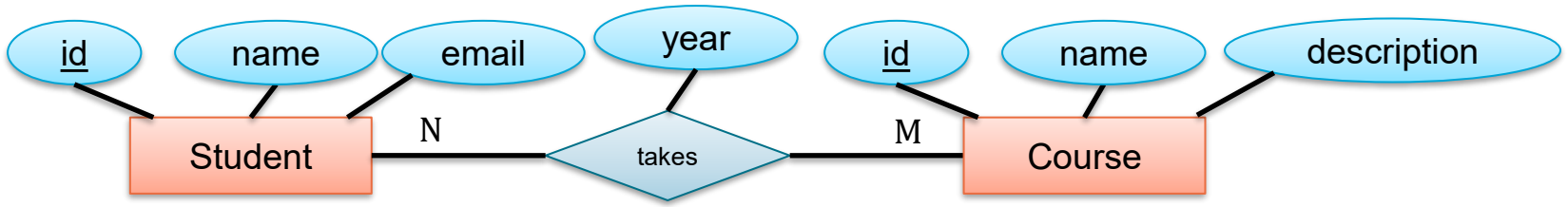
Logical Relational Schema
(Text Representation)

Logical Relational Schema
(ER Diagram)



Conceptual To Logical

- We can add additional attributes to relationships!

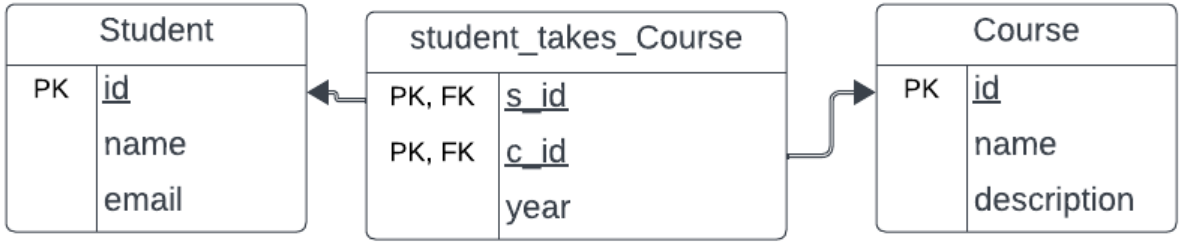


Conceptual Schema
(EER Diagram)

Student(id, name, email)
student_takes_course(s_id → Student(id), c_id→ Course(id), year)
Course(id, name, description)

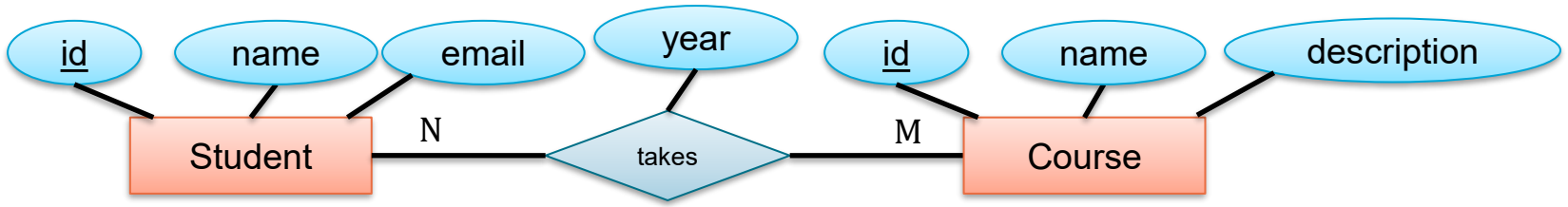
Logical Relational Schema
(Text Representation)

Logical Relational Schema
(ER Diagram)



Conceptual To Logical

- We can add additional attributes to relationships!
  - This does still not allow a student to take a course multiple times in different years
    - Primary Key is (Student.id, Course.id)!

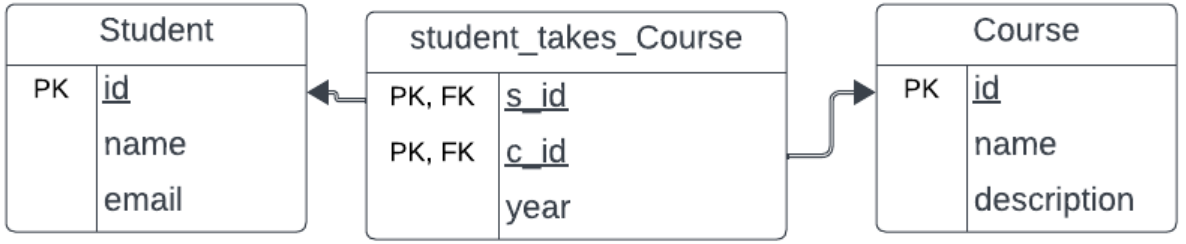


Conceptual Schema
(EER Diagram)

Student(id, name, email)
student_takes_course(s_id → Student(id), c_id→ Course(id), year)
Course(id, name, description)

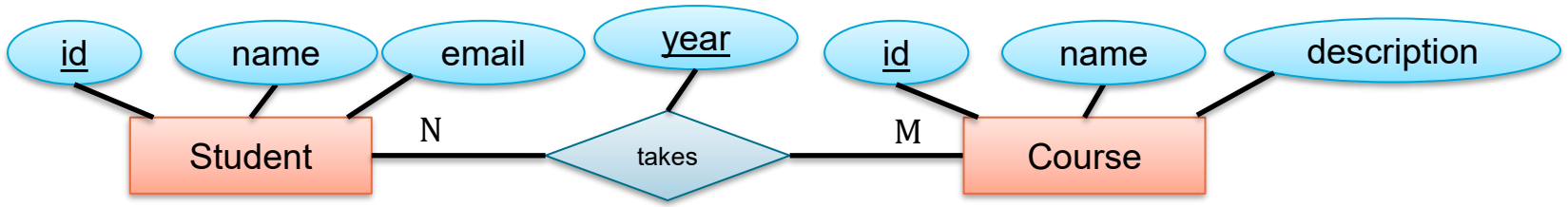
Logical Relational Schema
(Text Representation)

Logical Relational Schema
(ER Diagram)



Conceptual To Logical

- How can a student take a course multiple times?
  - Expand the primary key to include more attributes!
    - Taking a course is now identified by (Student.id, Course.id, year)!

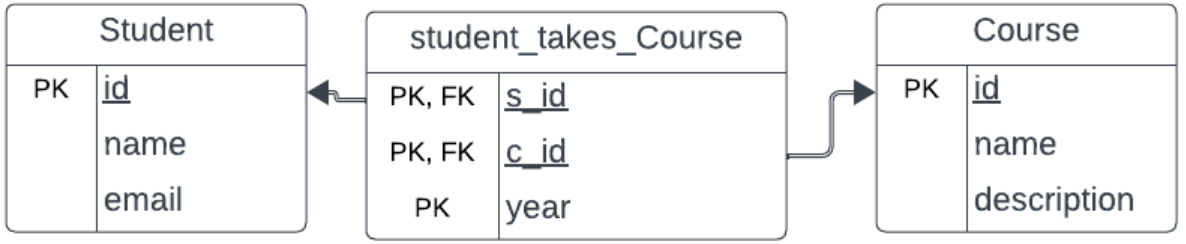


Conceptual Schema
(EER Diagram)

Student(id, name, email)
student_takes_course(s_id → Student(id), c_id→ Course(id), year)
Course(id, name, description)

Logical Relational Schema
(Text Representation)

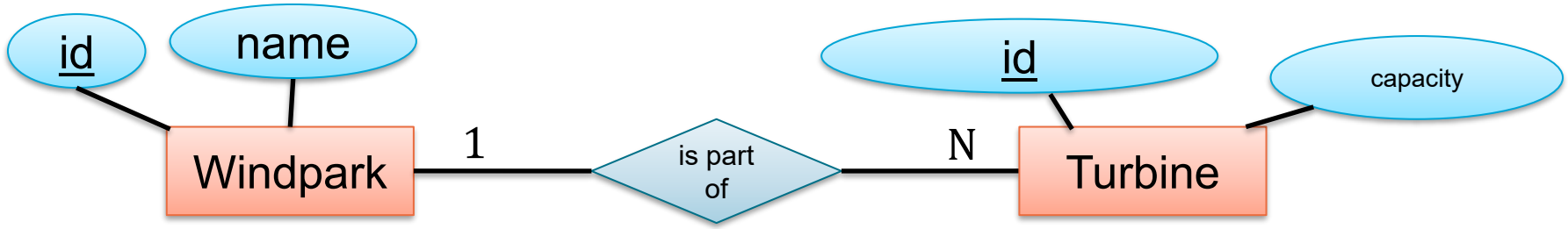
Logical Relational Schema
(ER Diagram)



Conceptual To Logical

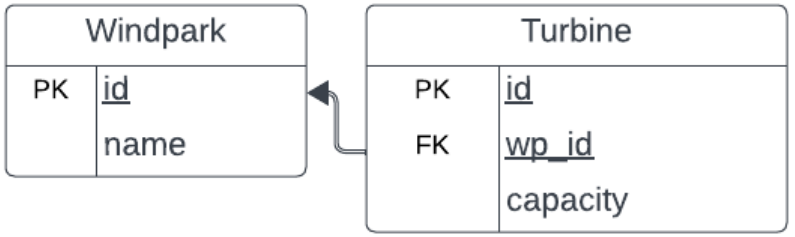
- Converting a 1:M relationship type a logical relation schema:
  - Entity Type at 1-side can only participate once at the relationship type
    - => Push relationship type to the 1-side

Conceptual Schema



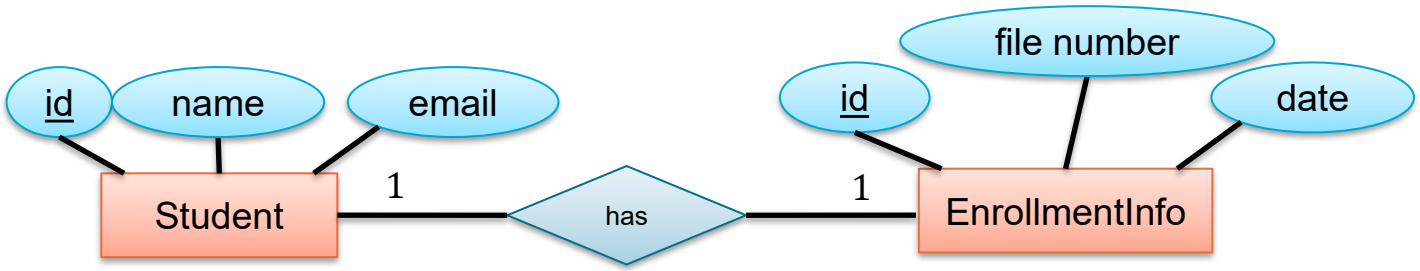
Logical (Relational) Schema

Windpark(id, name)
Turbine(id, capacity, **wp_id→ Windpark(id)**)



Conceptual To Logical

- Converting a 1:1 relationship type into a logical relation schema:
  - Many choices….
    - Variant 1: Treat as 1:M relationship
    - Variant 2: Merge into 1 table



Logical (Relational) Schema: Variant 1

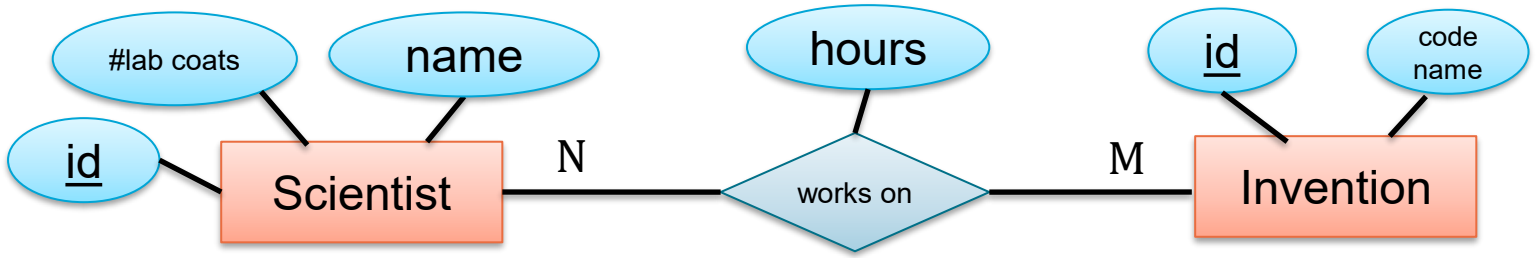
Student(id, name, email)
EnrollmentInfo(id, fileNumber, date, **s_id→ Student(id)**)

Logical (Relational) Schema: Variant 2

Student(id, name, email, enrollmentId, enrollmentFileNumber, enrollmentDate)

Conceptual To Logical

- How to deal with attributes attached to a relationship type
  - For N:M, just put:



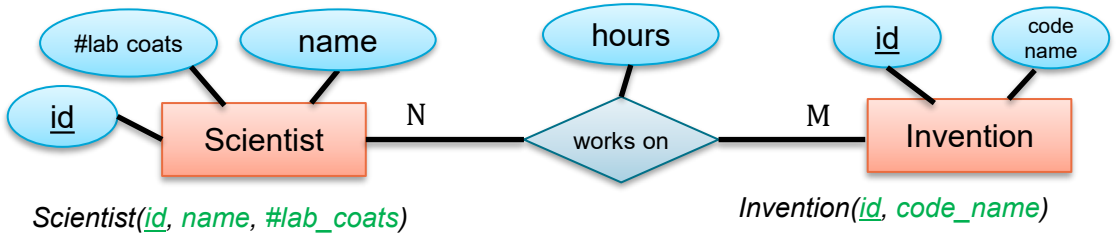
Scientist(id, name, #lab_coats)

scientist_works_on_invention(
scientist → Scientist(id),
invention → Invention(id),
hours
)

Invention(id, code_name)

Conceptual To Logical

- Don't do this!!

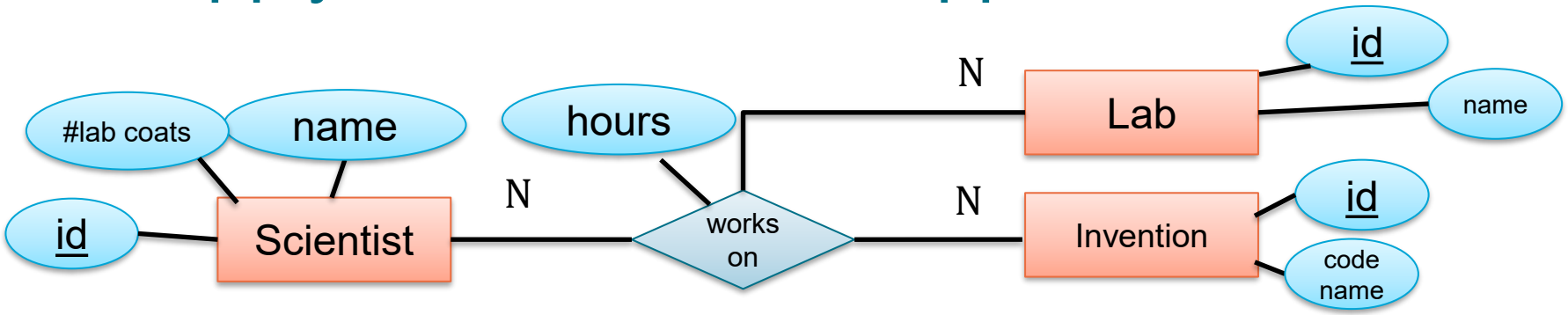


scientist_works_on_invention(
worksOnID,
scientist → Scientist(id),
invention → Invention(id),
hours
)

  - While this is not incorrect per se, the performance of this will be bad
    - Physical design heuristics (see IDM); messes up primary indexes

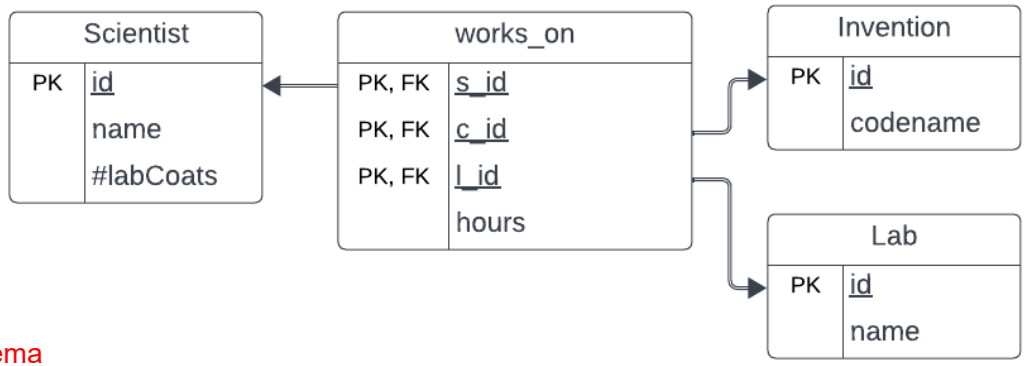
Conceptual To Logical

- What about n-ary relationship types? (n>2)
  - Just apply the exact same approaches:



Conceptual Schema
(EER Diagram)

works_on(
s_id → Scientist(id),
i_id → Invention(id),
l_id → Lab(id)
hours)



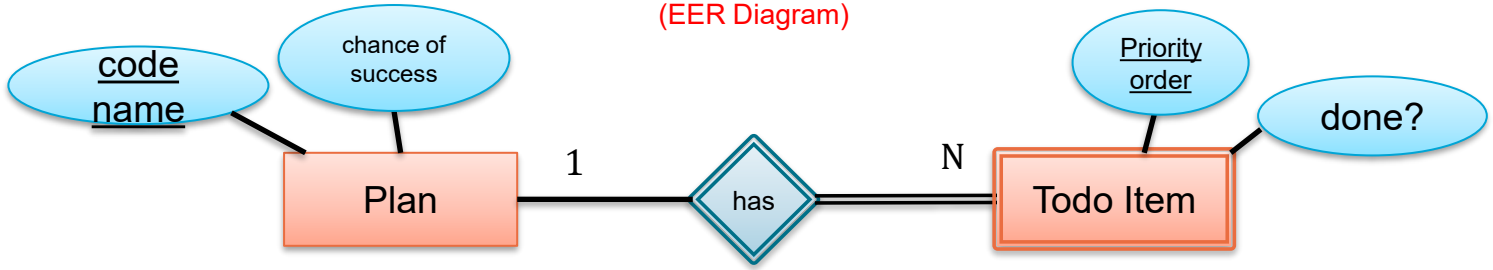
Logical Relational Schema

Conceptual To Logical

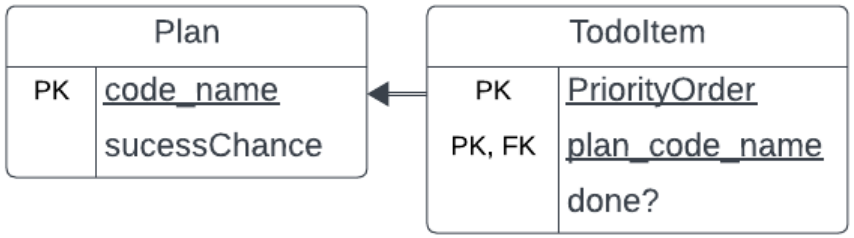
- Converting a weak entity into a relation schema:
  - Weak entities are only unique together with the entity at the **identifying relationship**

  => Follow identifying relationship and inherit respective foreign keys

Conceptual Schema
(EER Diagram)

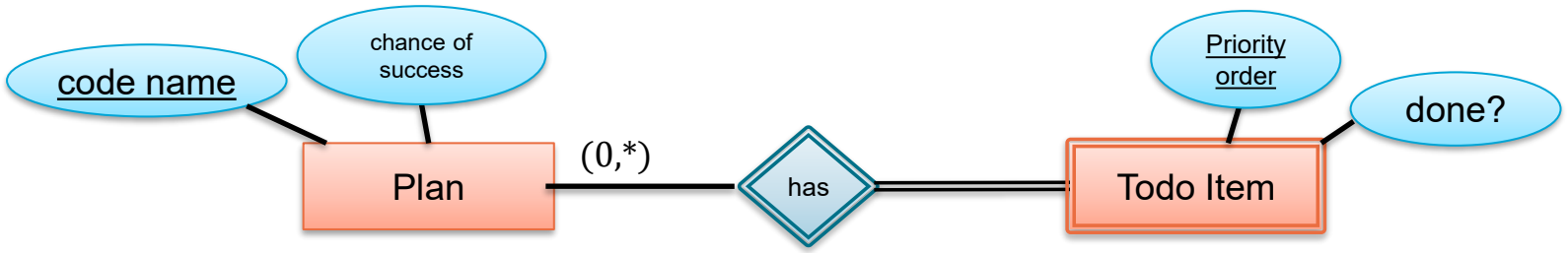


Logical Relational Schema



Evil_Plan(*code_name*, *chance_of_success*)
todo_item(priority_order, plan → Plan(codename), done)

Conceptual To Logical



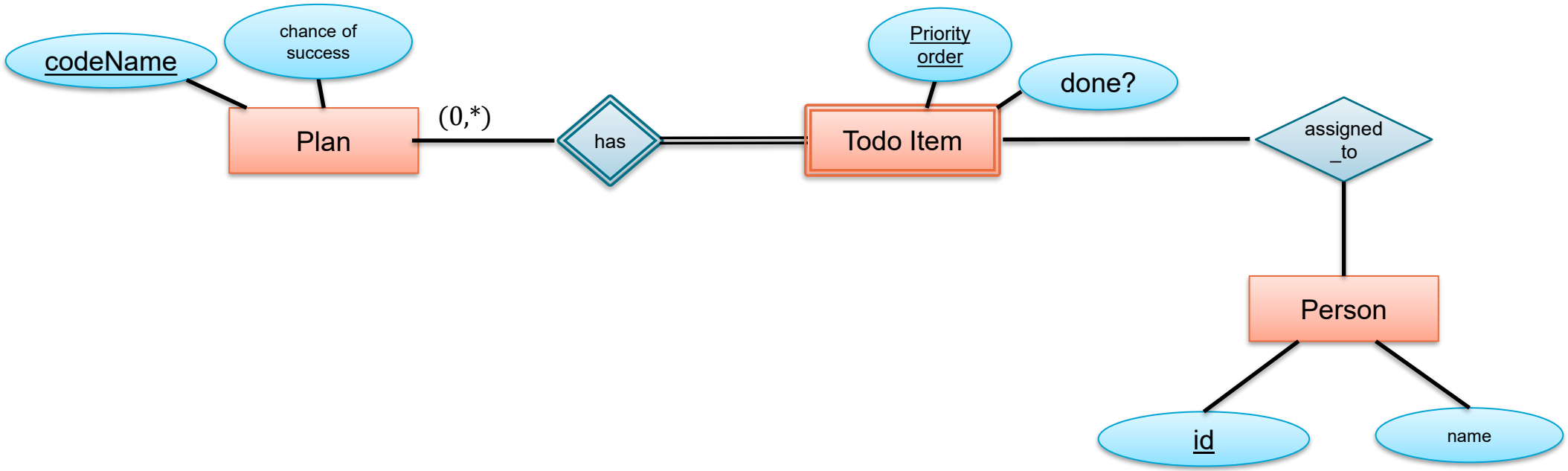
Plan(*code_name*, *chance_of_success*)
todo_item(priority_order, plan_name → Plan(code_name), done)

Vs:

Evil_Plan(*code_name*, *chance_of_success*)
todo_item(plan_name → Plan(code_name), priority_order, done)

- Spoiler IDM CSE1505: "2nd version is likely better as it will order all todo items on disc by evil plans; so, all todo items belong to a plan are close by
  - This is a bit of physical design…

Conceptual To Logical



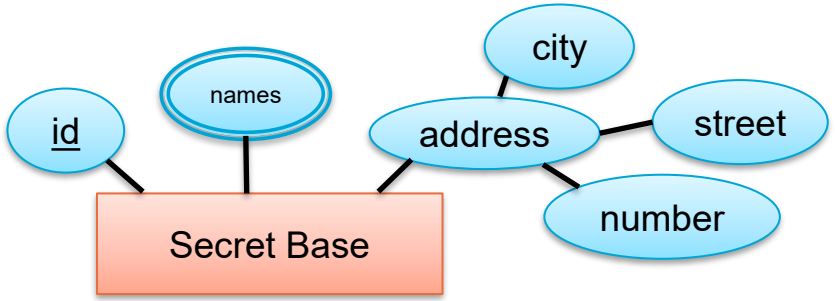
Plan(*codeName*, *chanceOfSuccess*)
todo_item(priorityOrder, planName → Plan(codeName), done)
assigned_to(priorityOrder, planName, personId→ Person(id), **(priorityOrder, planName) → todo_Item(priorityOrder, planName))**
Person(id, name)

Foreign key constraint to todo_item

- The foreign Key Constraint in assigned_to towards Todo_Item needs to involve the whole primary key, e.g., both priotityOrder and planName.
  - We denote that somewhat similar to how SQL works, see bold BLUE highlight

Conceptual To Logical

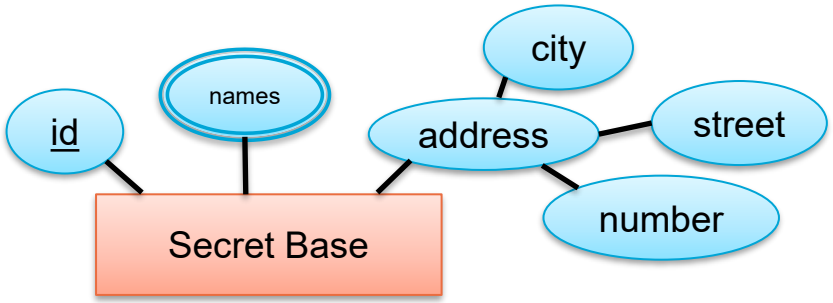
- How to deal with multi-attributes and composite attributes
  - **Composition:** just flatten it
  - **Multi-attribute:** treat it like a weak entity



Version 1:
Secret_Base(id, addr_city, addr_street, addr_number)
base_name (hideout → Secret_Base, name)

(Own relation for names, adress attributes flattened into Secet_Base)

Conceptual To Logical



Version 1:
Secret_Base(id, addr_city, addr_street, addr_number)
base_name (hideout → Secret_Base, name)

(Own relation for names, adress attributes flattened into Secret_Base)

Version 2:
Secret_Base(id)
base_name (hideout → Secret_Base, name)
adresses(city, street, number, base → Secret_Base)

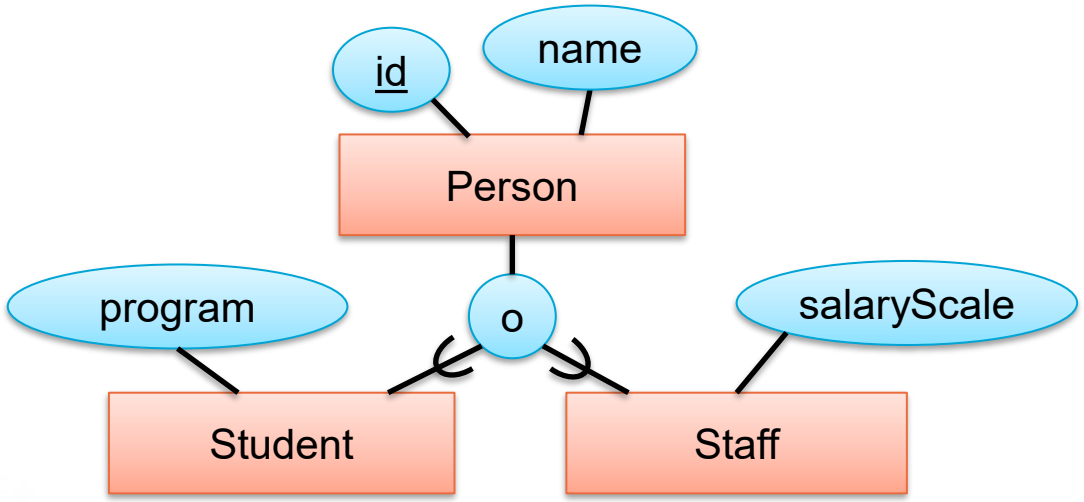
(Own relation for names and address; this basically represents now a 1:M relationship between bases and addresses so likely not as good as version 1?)

Version 3:
Secret_Base(id, addr_city, addr_street, addr_number, name1, name2, name3, name4, name5, name6, name7)

(Mhh….☹ Maybe. Depends. In most cases this is probably a very bad solution….- could even argue that it's wrong because what happens if something has 8 names? Still, people do this.)

Conceptual To Logical

- Converting types with inherited attributes/ relations into a relation schema:
  - Can be implemented in many ways
    - More than 5…

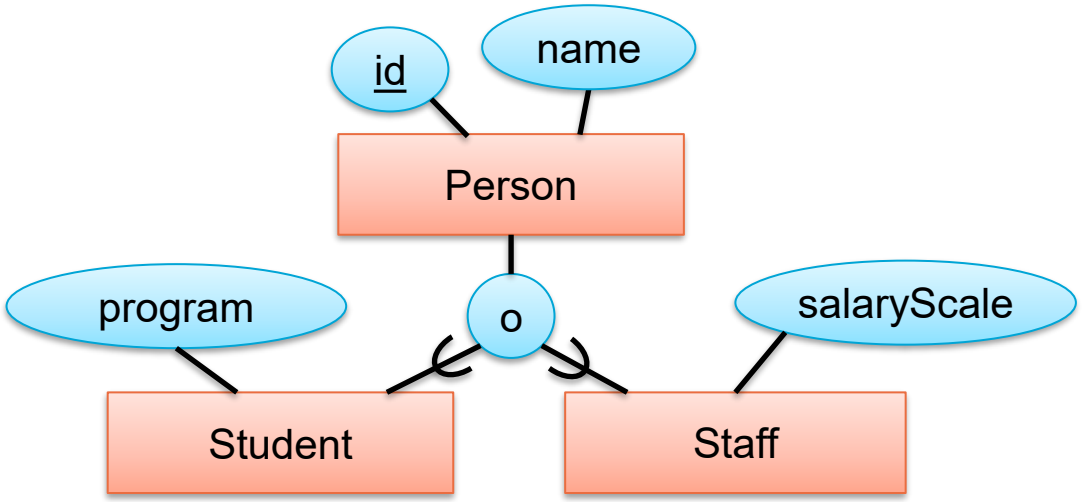


Conceptual To Logical

Overlapping non-total Inheritance:
Persons can be Students, or Staff, or both or neither.

- Version 1:

Person(id, name)
Student(id → Person(id), program)
Staff(id → Person(id), salaryScale)

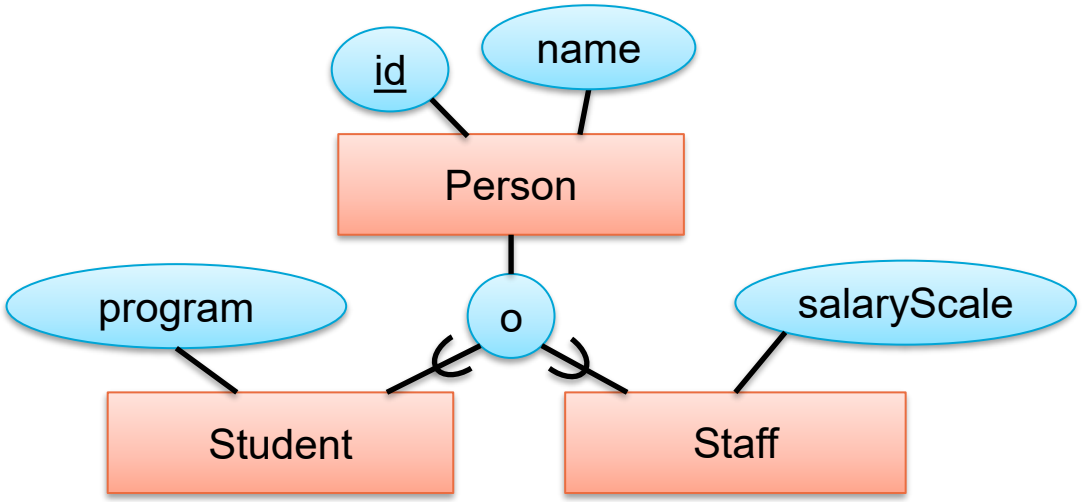


All Persons (including Students and Staff) have a row in **Person**,
Students have an extra row in **Student** referring to Person,
Staff have an extra row in **Staff** referring to Person
-> Clean but maybe clumsy?

Conceptual To Logical

Overlapping non-total Inheritance:
Persons can be Students, or Staff, or both or neither.

- Inheritance Version 2:



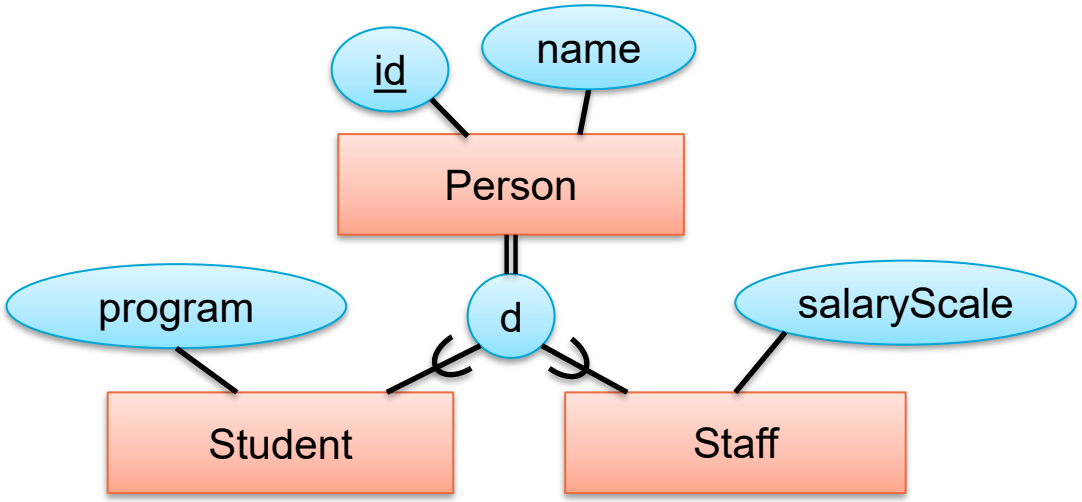
Person(id, name)
Student(id, name, program)
Staff(id, name, salaryScale)

Students have a row in Student table.
Staff have a row in Staff table.
Students who are also Staff have a row in both **Student and Staff**.
Person who are neither Students nor Staff have a row in **Person**.
-> Likely not a nice solution?

Conceptual To Logical

Disjoint total Inheritance:
Persons are Students or Staff, but not both nor neither.

- Inheritance Version 2: (now total disjoint!!)



Student(id, name, program)
Staff(id, name, salaryScale)

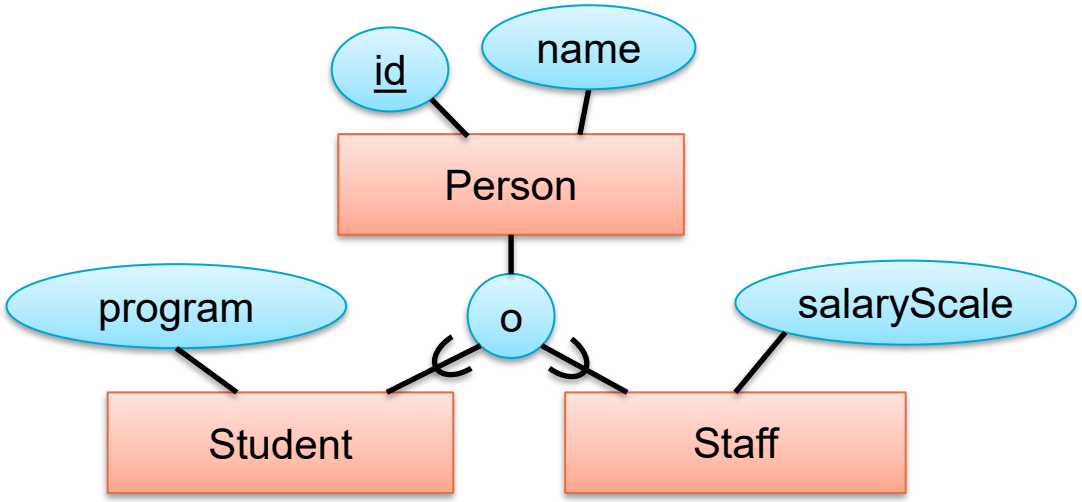
No need for Persons anymore (as everything is either Student xor staff)
Potential mess with ids (there could be a student with same id as a staff member), but likely okey?
-> Better…?

Conceptual To Logical

Overlapping Non-Total Inheritance:
Persons can be Students, or Staff, or both or neither.

- Version 3:

Person(<u>id</u>, name, program, salaryScale)



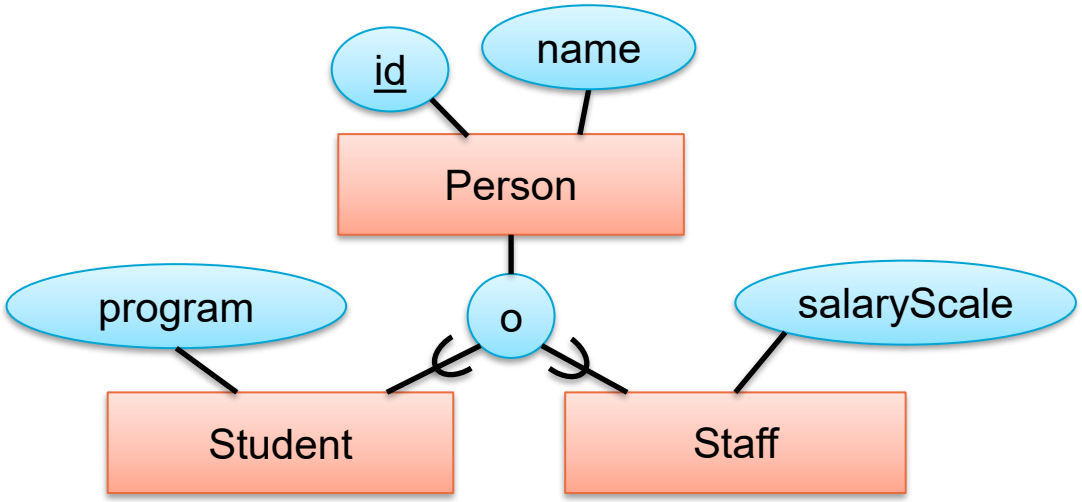
Only one big table!
Use program attribute iff Student, use salaryScale iff Staff.
Leave unused attributes empty/NULL.
No explicit type information in the logical schema.
It's somewhat ugly but also popular.

Conceptual To Logical

Overlapping Non-Total Inheritance:
Persons can be Students, or Staff, or both or neither.

- Version 3b:

Person(id, type, name, program, salaryScale)



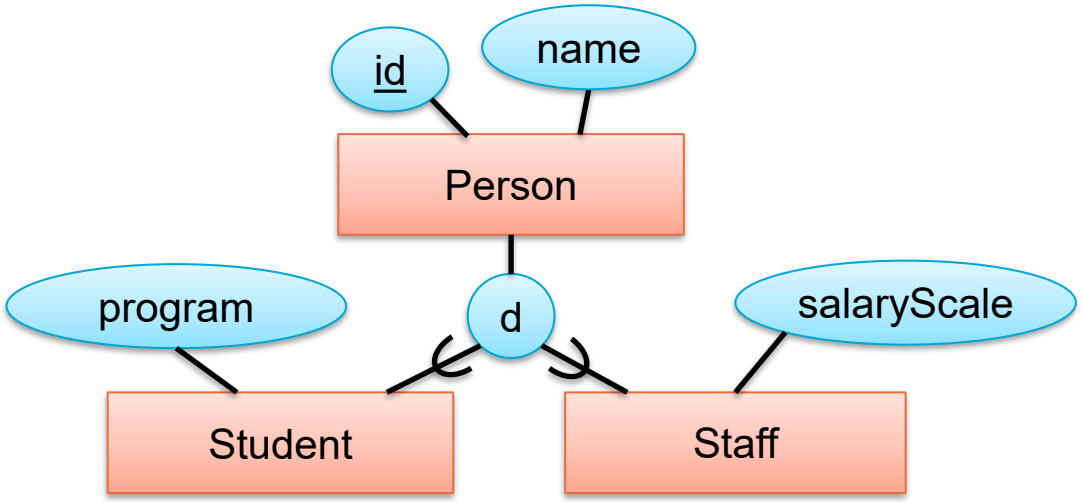
A variant of Version 3: Just add a new “type” attribute which stores what type a person is, e.g. “student”, “staff”, “both”, “neither”. Checking subtypes involves a lot of manual String comparisons.

Debatable if this is nice…

Conceptual To Logical

Disjoint Non-Total Inheritance:
Persons can be Students, or Staff, or neither.

- Version 3c:



Person(id, type->Type(tid), name, program, salaryScale)
Type(tid, type_name)

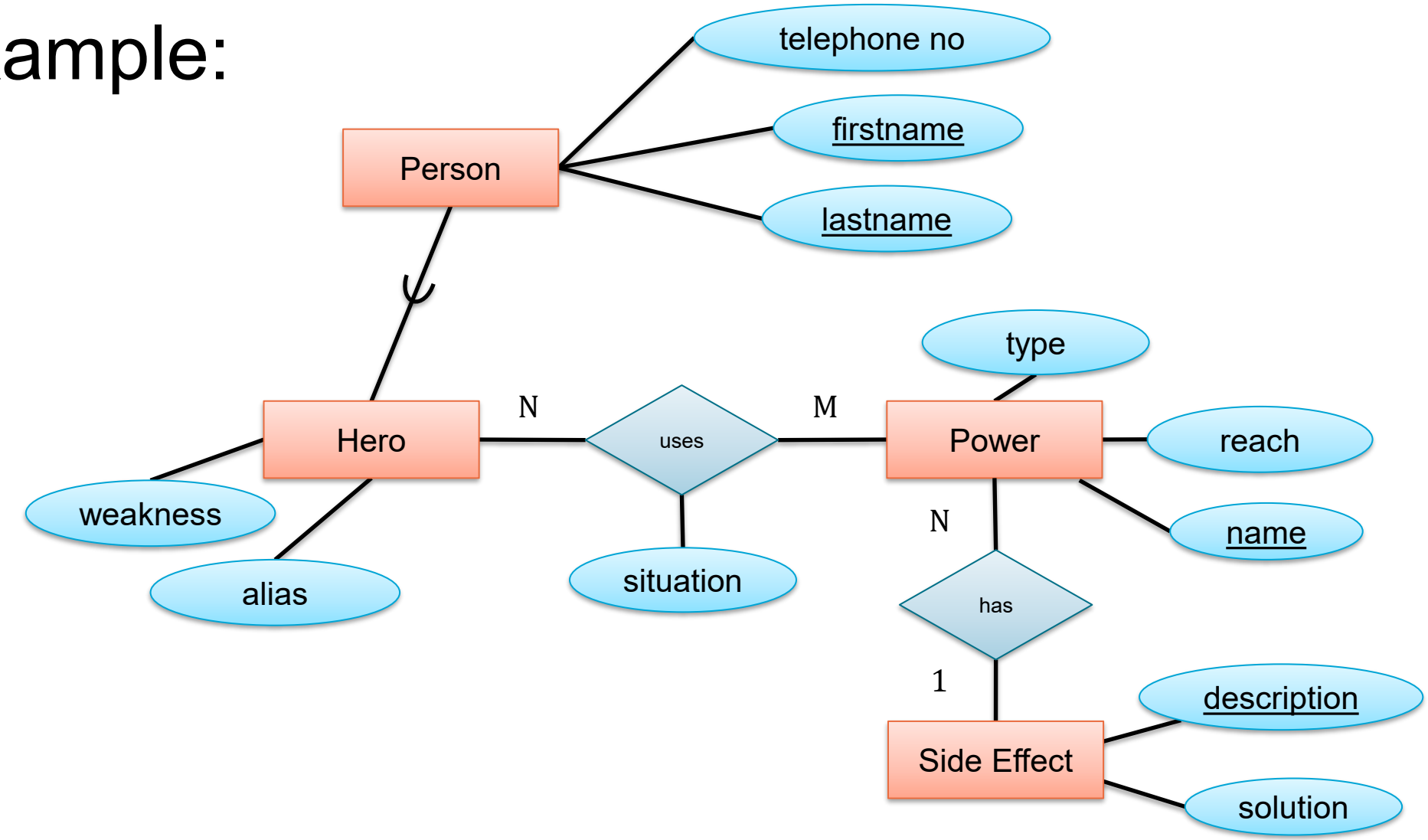
A variant of Version 3b: A second table holds information about all available types, e.g. “staff”, “student”, etc. Works only nicely for disjoint inheritance.

Conceptual To Logical

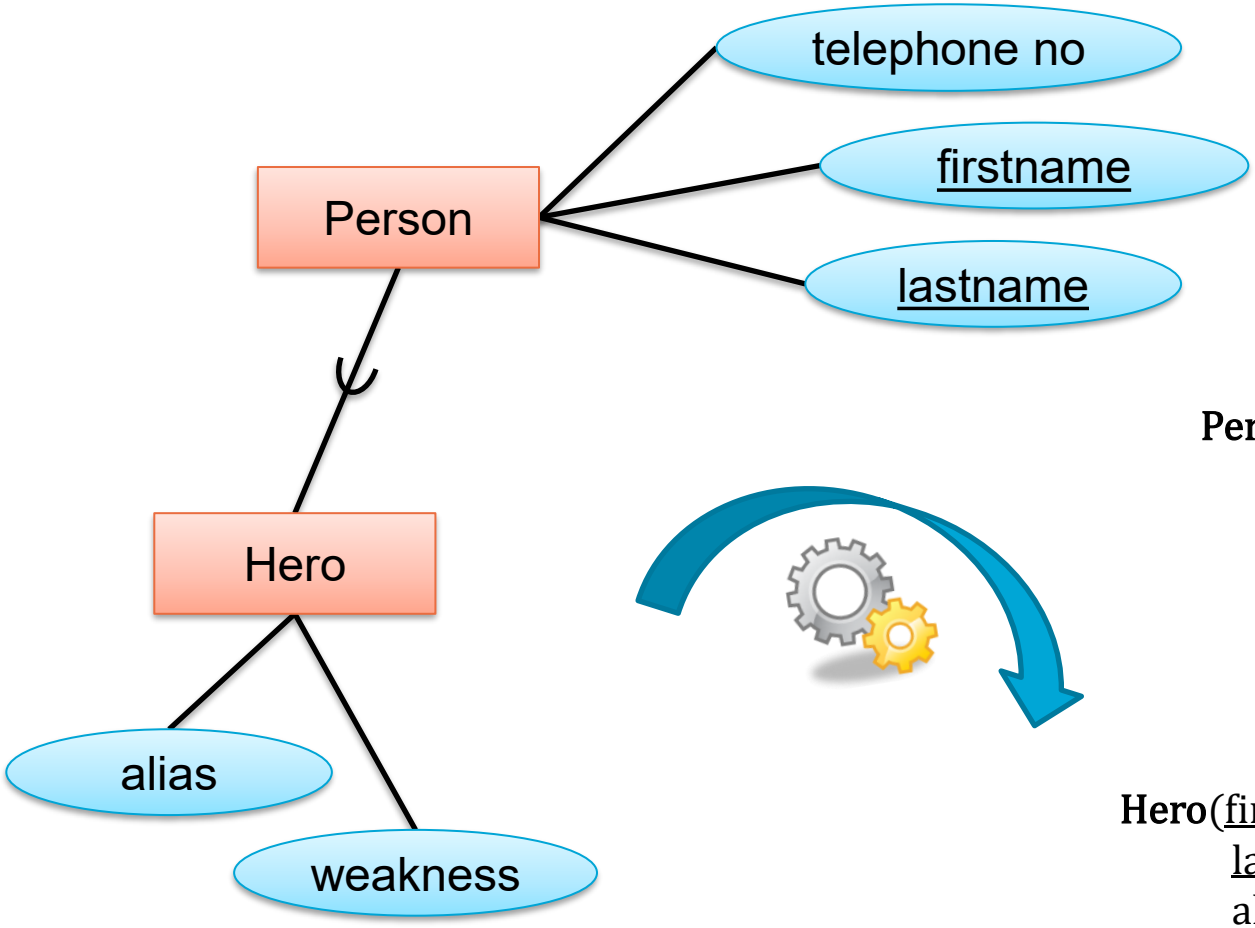
- There are many more variants of how to implement inheritance in a logical relational model than those discussed here…

Example: Conceptual To Logical

- Example:



Example: Conceptual To Logical



Person(<u>firstname</u>,
<u>lastname</u>,
telephone_no)

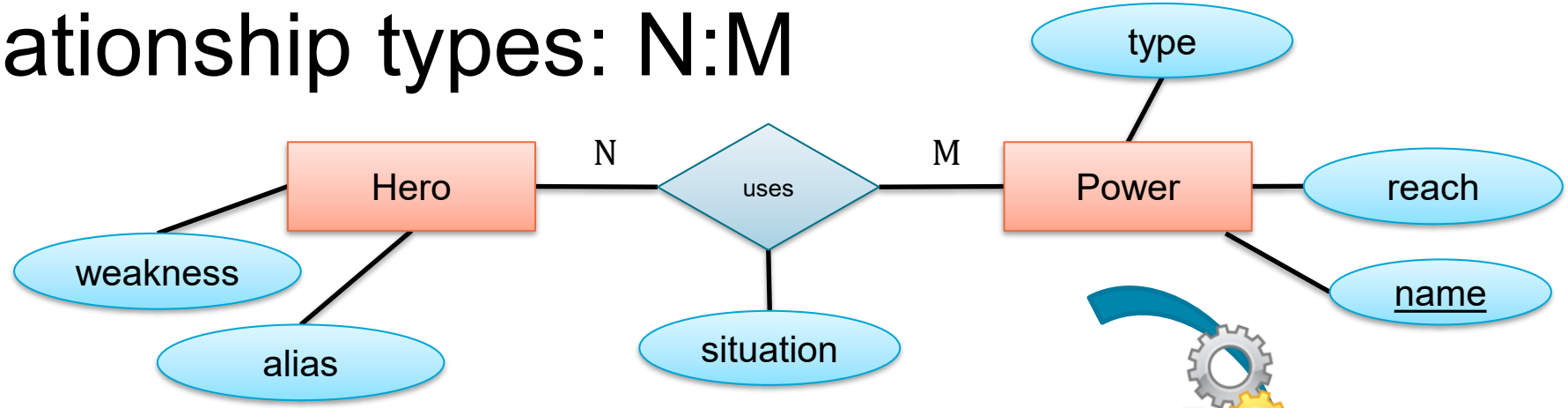
Hero(<u>firstname</u>,
<u>lastname</u>,
alias,
weakness,
(firstname, lastname) → Person(firstname, lastname)
)

Quick Explanation of Previous Slide

- Person has a composite primary key (Firstname, LastName). So, the PK is 2 columns. Lets say we have two persons: (Diana, Prince) (Clark, Kent)
- Hero refers to Persons. Thus, we have the schema
  - A) Hero(firstname, lastname, weakness, (firstname, lastname)-> Person(Firstname, Lastname))
  - Note that (firstname, lastname)-> Person(Firstname, Lastname) is the definition of a constraint, NOT a column.
- Why is it it important to define the column like this, and not as
  - B) Hero(firstname->Person(Firstname), lastname->Person(Lastname), ...) ??
  - Look at this example of Heroes: (Clark, Prince, Wonderman, Aphroditonite) (Diana, Kent, Superwoman, Kryptolaw)
  - Clearly, these two tuples would be valid in schema B, eventhough they do not refer to a real person. Thus, only schema A does this properly.

Example: Conceptual To Logical

- Relationship types: N:M

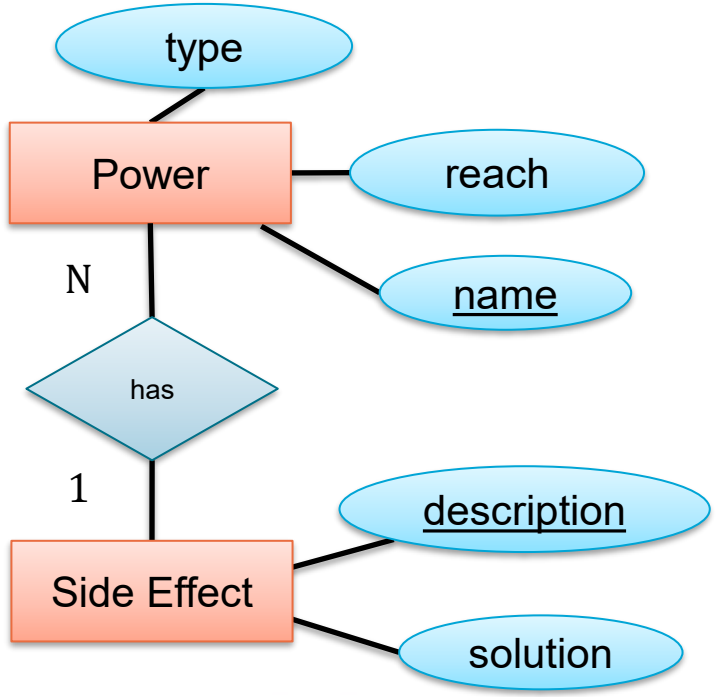


Hero(<u>firstname</u>,
<u>lastname</u>,
alias,
weakness,
(firstname, lastname) → Person(firstname, lastname)
)

Power(<u>name</u>,
type,
reach)

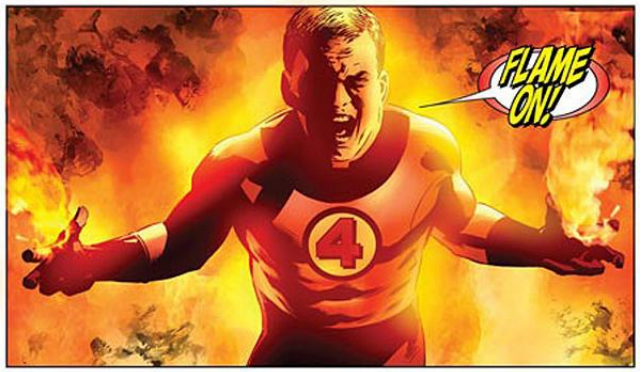
uses(<u>firstname</u>,
<u>lastname</u>,
<u>name</u> → Power(name),
situation,
(firstname, lastname)
→ Hero(firstname, lastname)
)

Example: Conceptual To Logical



Power(<u>name</u>,
type,
reach,
side_effect→ SideEffect(description))

SideEffect(<u>description</u>,
solution)

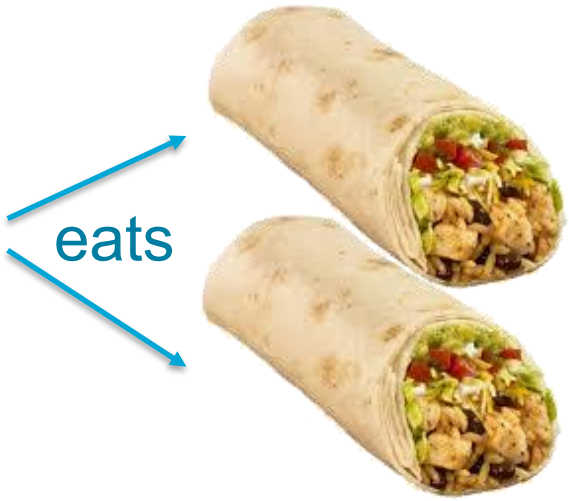


Some EER Discussions

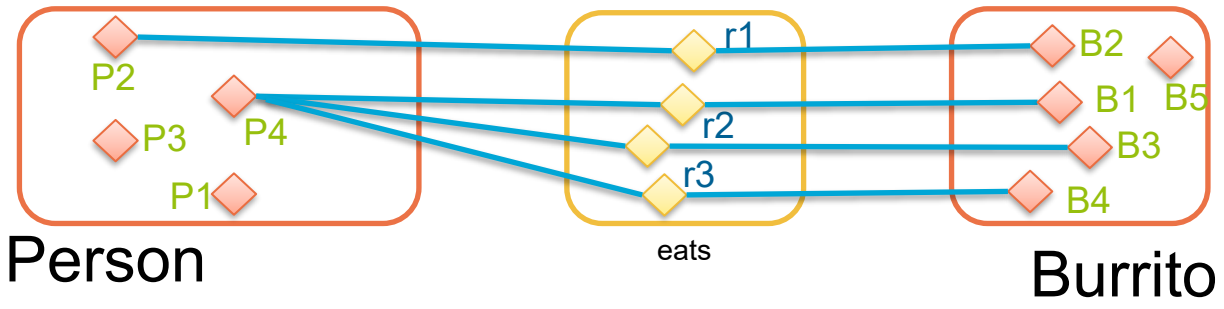
This is just for clarification, read at home!
Don’t worry about this too much; it’s not exam topics.

Cardinal Nightmare

- "How many instances of type X are in relationship with instances of type Y??"
  - "Does Christos eat multiple burritos, or does a burrito eat multiple Christos?"

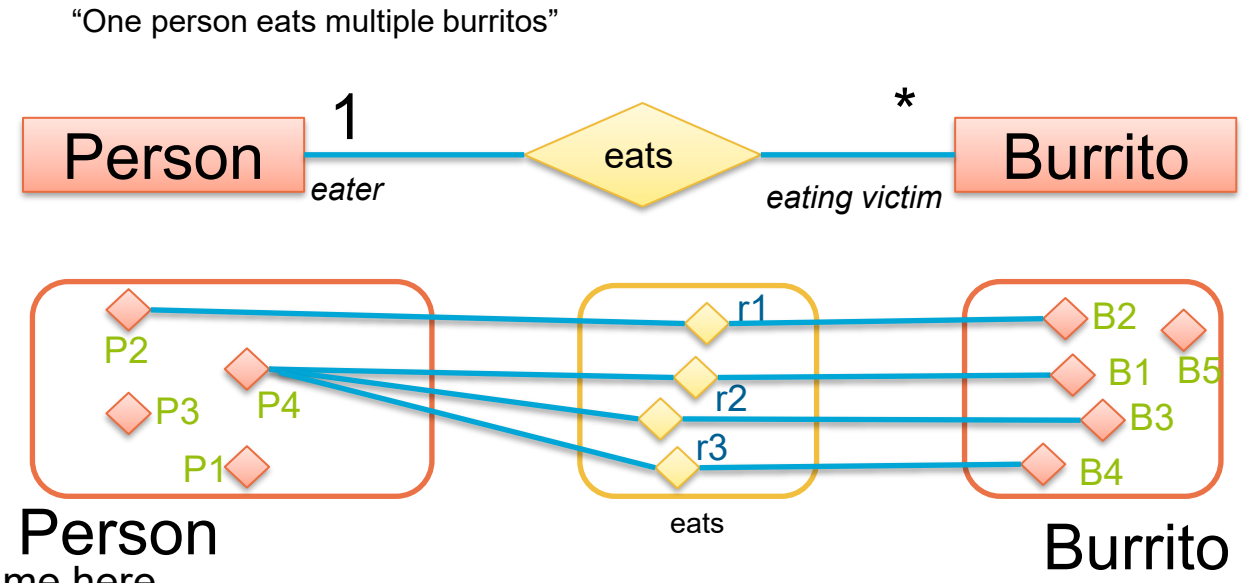


Cardinal Nightmare



Cardinal Nightmare: UML Style

- **UML Style** (our course's default):
  - Usually only distinguishes between 1, * or (0, 1) or (1, *)
    - Fine-granular cardinalities and higher arity relations not easily possible
      - » * is sometimes written N



"N" and "*" means the same here

Cardinal Nightmare: UML Style

"One person eats multiple burritos."
(but there can be burritos being left over, or people who don't eat anything, or just 1, or 10)

"Multiple persons share one burrito."
(but there can be burritos being left over, or people who don't eat anything)

"People eat burritos."
(Persons can share burritos, eat them by themselves, leave some over, or not eat anything)

"One person eats multiple burritos."
(each burrito must be eaten, but some people can choose to not eat anything)

"N" and "*" means the same here

Cardinal Nightmare: UML Style

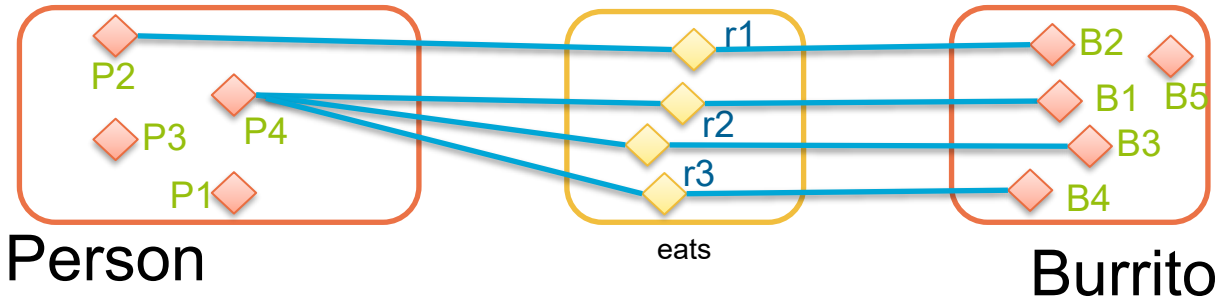
- UML style can also provide min / max
  - but only uses 0, 1, N/*
  - "1" stands for "(0,1)": up to one
  - "N" stands for "(0,N)": any number of
  - "(1,1)": exactly 1
    - which is the same as 1 with total participation…
  - "(1,N)": at least 1
    - Which is the same as N with total participation

"Each person eats exactly one burrito, and each burrito is being eaten"

Cardinal Nightmare: UML style

- **UML Style** (our course's default):
  - Usually only distinguishes between 0, (0, 1),1, (0, *) and *
    - Fine-granular cardinalities and higher arity relations not easily possible

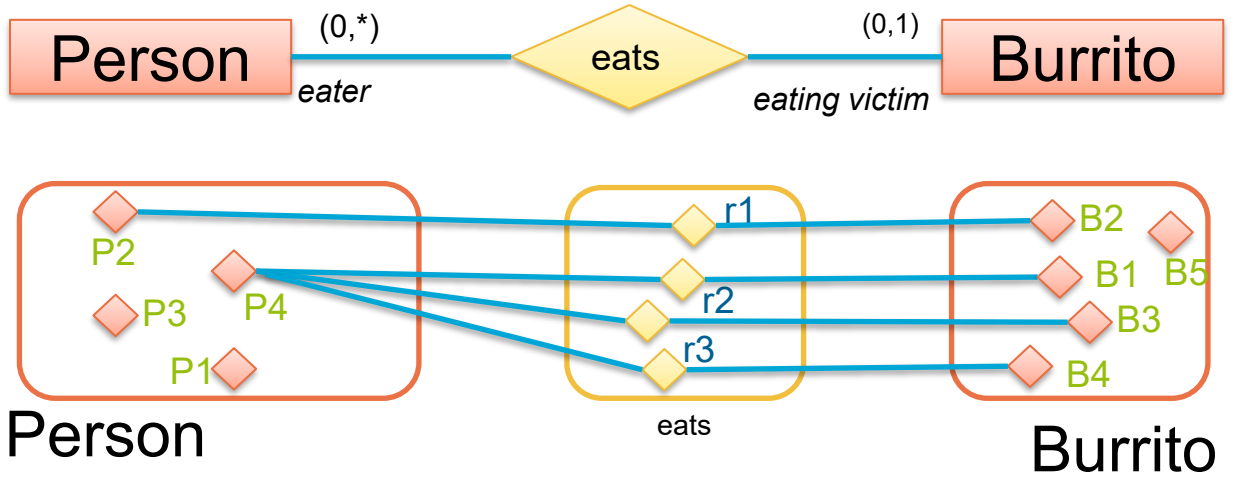
USE THIS IN THE EXAM



Cardinal Nightmare: Classic Style

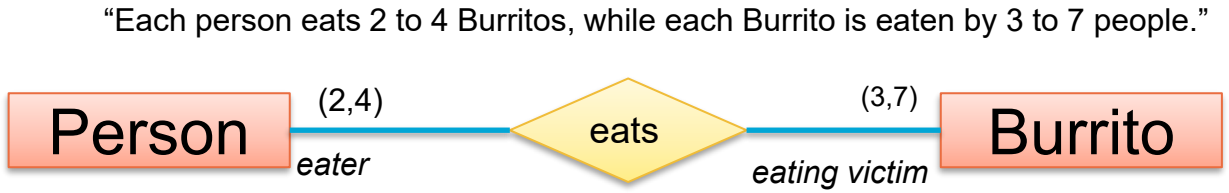
- Mistake: **Confusing Cardinality Notations**
  - Warning: There are several conflicting ways of writing cardinalities!
    - Whichever way you choose, **BE CONSISTENT!**
  - **Classic ER Style (not default in this course but commonly found)**:
    - This notation is needed for non-trivial n-ary relationships (n > 2)

"Each person eats 0 to any number of Burritos, while each Burrito is eaten by 0 to 1 people."

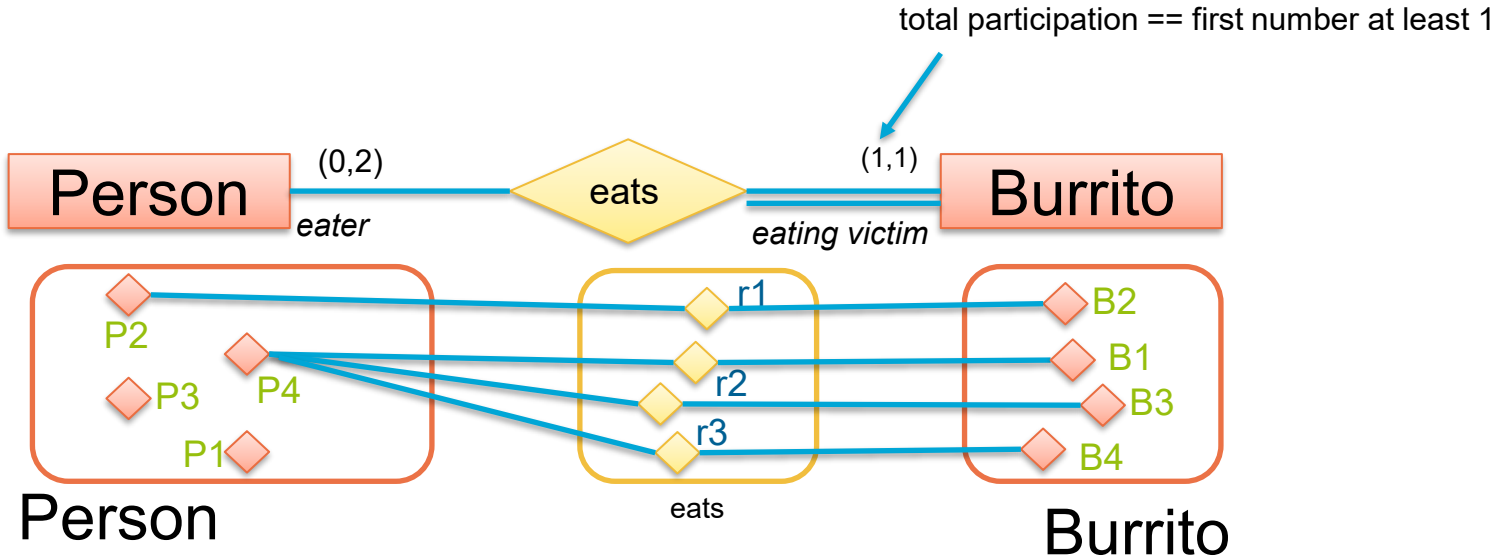


Cardinal Nightmare: Classic Style

- Does allow very precise cardinalities



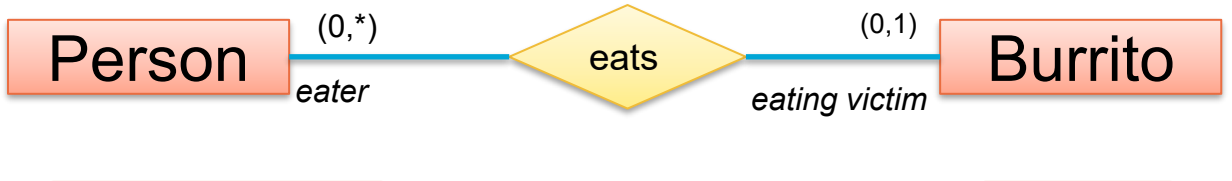
- This is where the "total participation" comes from

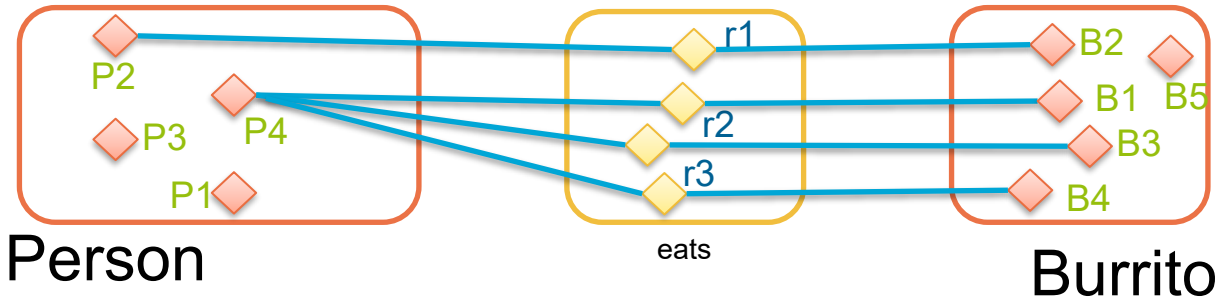


Cardinal Nightmare: Classic Style

- Mistake: **Confusing Cardinality Notations**
  - Warning: There are several conflicting ways of writing cardinalities!
    - Whichever way you choose, **BE CONSISTENT!**
  - **Clas** **e but com**
    - T **nships (n > 2)**

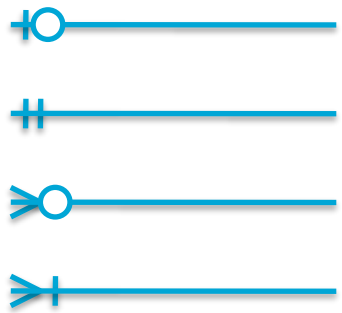
DO NOT USE THIS IN THE EXAM!





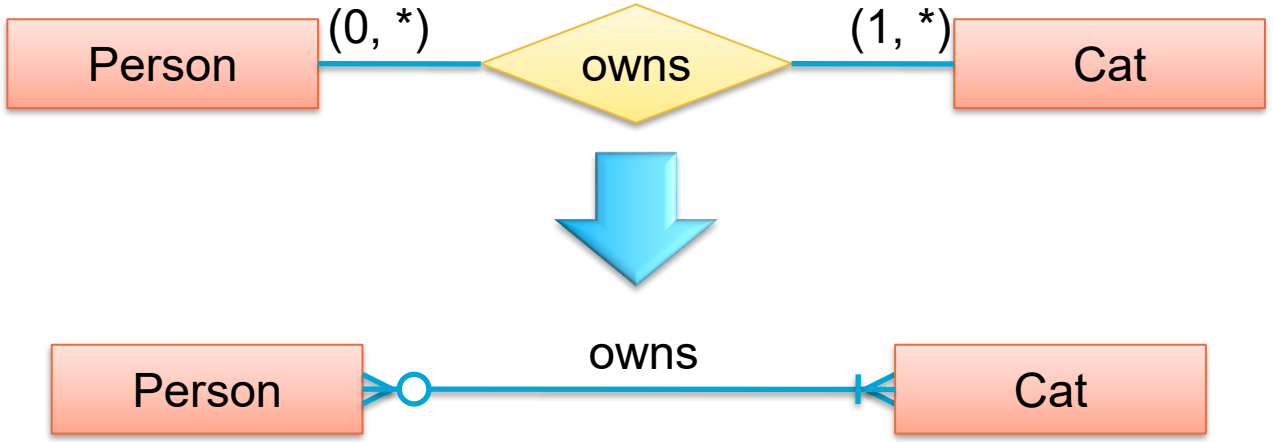
ER - Crow's Foot Notation

- **Many** derivates of Chen ER became popular
- **Crow's Foot Notation: Relationship Types**
  - Relationship types are modeled by lines connecting the entities (no explicit symbol for relationships)
  - The line is annotated with the name of the relationship which is a verb
  - Cardinalities are represented graphically
    - **(0, 1):** zero or one
    - **(1, 1):** exactly one
    - **(0, *):** zero or more
    - **(1, *):** one or more



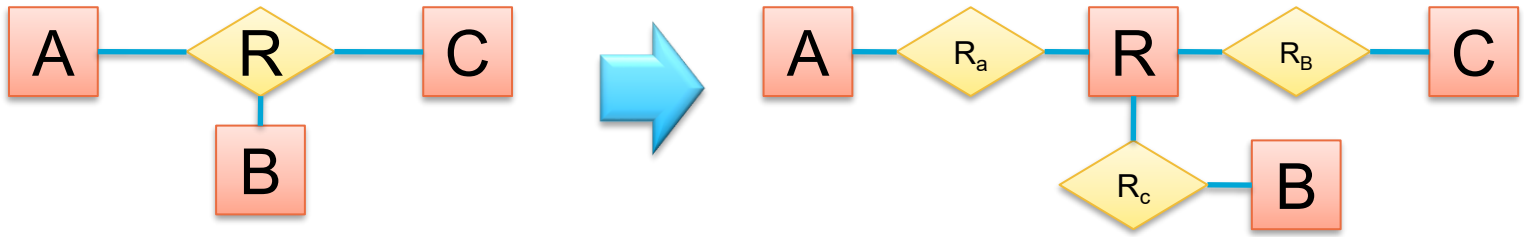
ER – Crow's Foot Notation

- Attention:
  - Cardinalities are written in UML style (and not Chen notation)

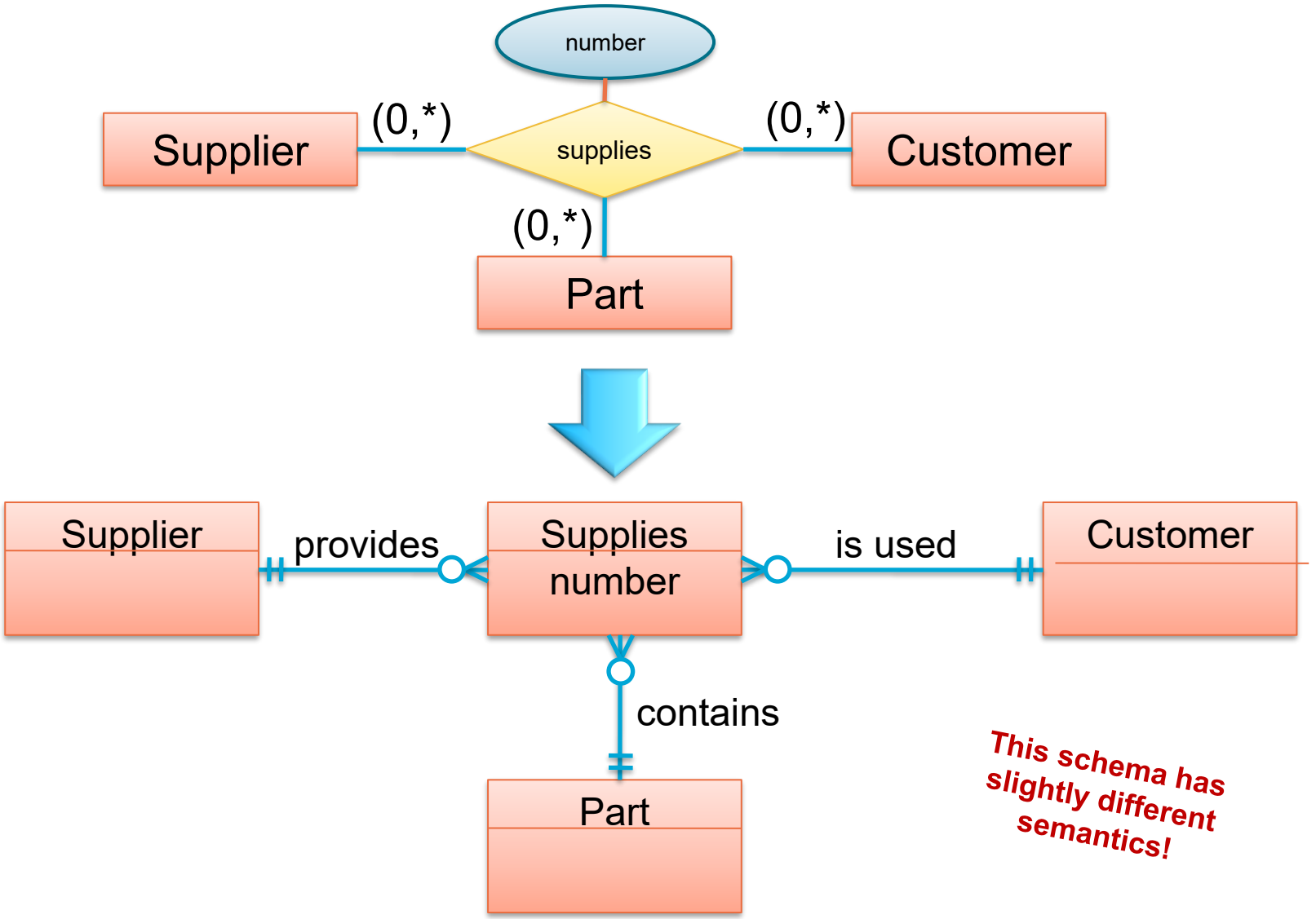


ER – Crow's Foot Notation

- **Problem**
  - N-ary relationship types **are not supported** by Crow's Foot notation, neither are relationship attributes
- **Workaround solution**:
  - **Intermediate entities** must be used
    - N-ary relationships are broken down in a series of **binary** relationship types anchoring on the intermediate entity



ER – Crow's Foot Notation

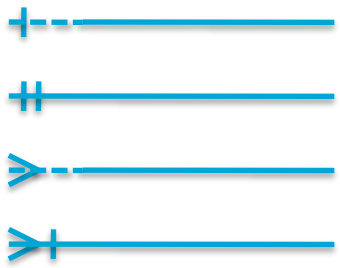


ER – Crow's Foot Notation

- Originally, ER diagrams were intended to be used on a **conceptual** level
  - Model data in an abstract fashion **independent** of implementation
- Crow's Foot notation sacrifices some conceptual expressiveness
  - The model is closer to the **logical** model (i.e., the way the data is later really stored in a system)
  - This is **not** always **desirable** and may obfuscate the intended semantics of the model

ER – Even more notations…

- **Barker's notation**
  - Based on Crow's Foot notation
  - Developed by Richard Barker for **Oracle**'s CASE modeling books and tools in 1986
  - Cardinalities are represented differently
    - **(0, 1):** zero or one
    - **(1, 1):** exactly one
    - **(0, N):** zero or more
    - **(1, N):** one or more

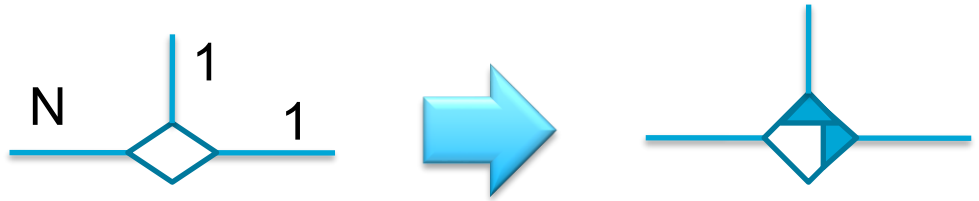


    - Cardinalities position similar to Crow's Foot notation and opposite to classic ER
  - Different notation of subtypes

ER – Even more notations…

- **Black Diamond Notation**
  - Cardinalities are represented differently
    - Cardinality annotation per relationship, not per relationship end
    - 1:1
    - 1:N
    - N:M

  - Also, N-ary relationships possible
    - Ternary

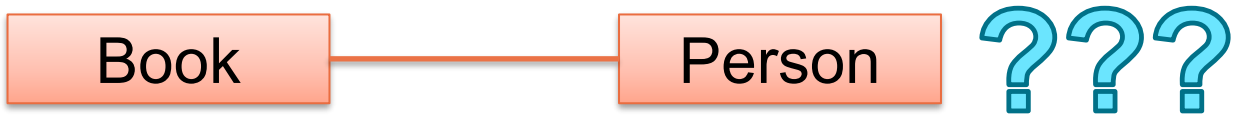


Common Modelling Mistakes

TU Delft WIS Web Information Systems

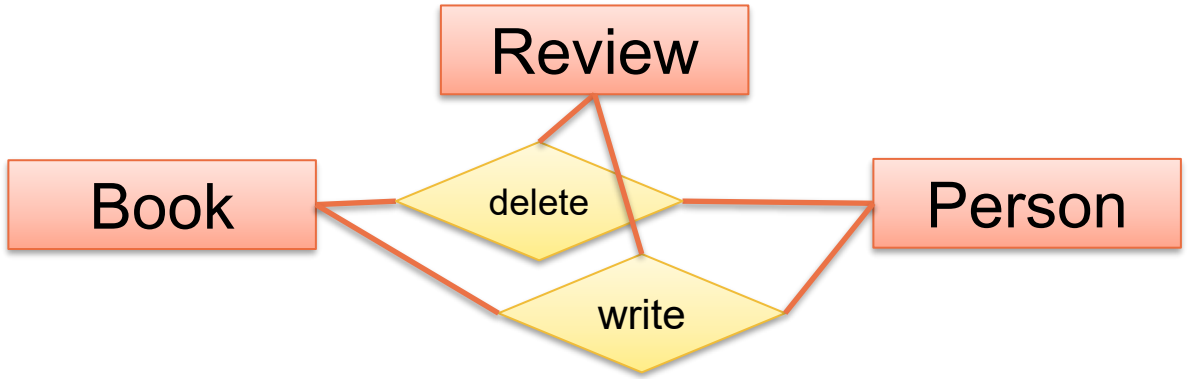
Common Mistakes in ER-Modelling

- Mistake: **No Primary Key**
  - Just don't do that outside of simple examples
  - (Same is true for cardinalities)
- Mistake: **No Relation Symbol or Name**



Common Mistakes in ER-Modelling

- Mistake: **Modelling Functionality as Data**
  - "Users can write reviews for a book, which they can also delete."



???

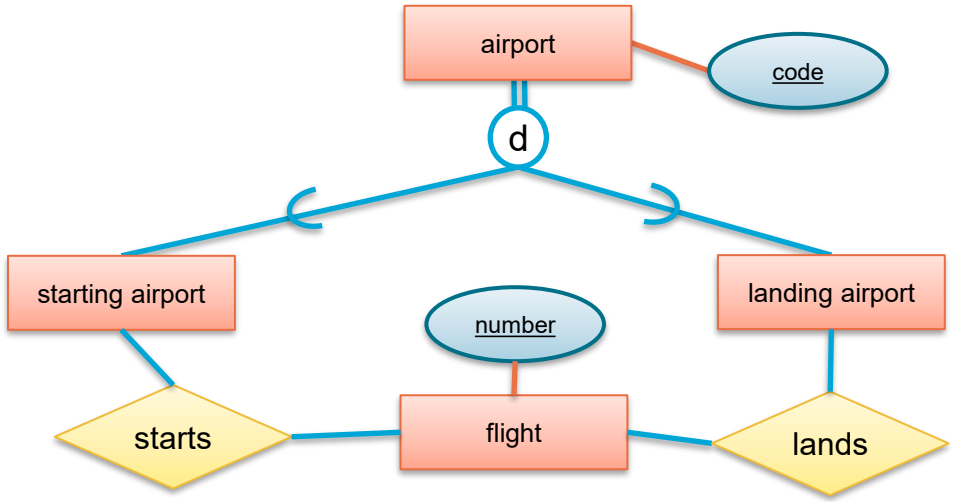
  - "Reviews can be posted on Facebook."

???

Common Mistakes in ER-Modelling

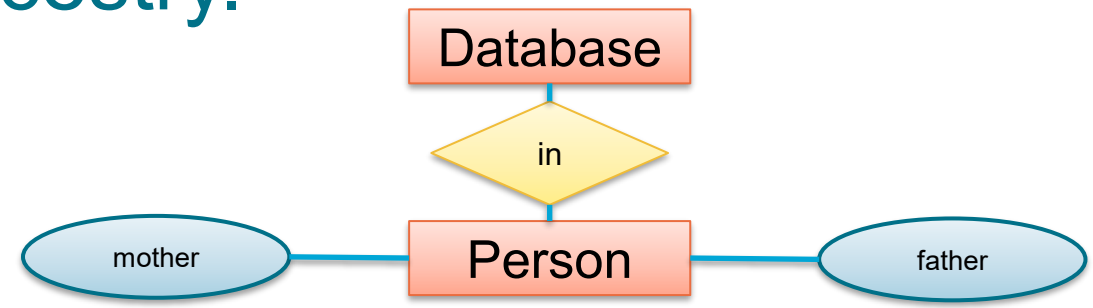
- Being completely confused
  - "Airports are in cities and have an airport code. Flights have a flight number and go from a starting airport to a landing airport."

???

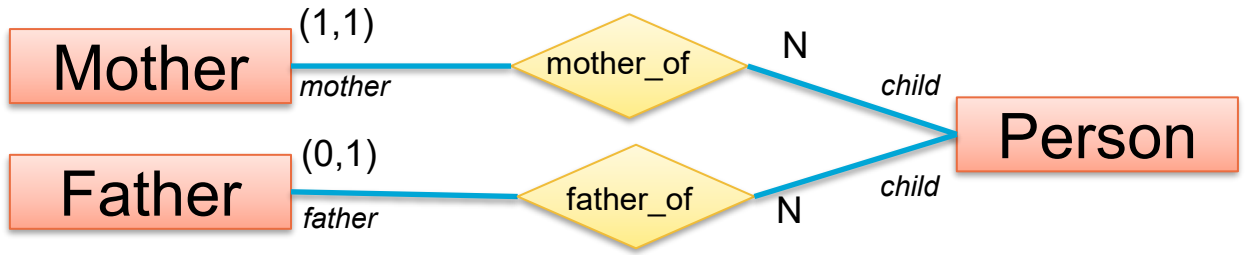


Common Mistakes in Modelling

- Mistake: **Model not suitable for the task**
  - Task: “Build a person database. Each person should have a mother and a father. The database is later used to explore ancestry. ”
  - Very Bad

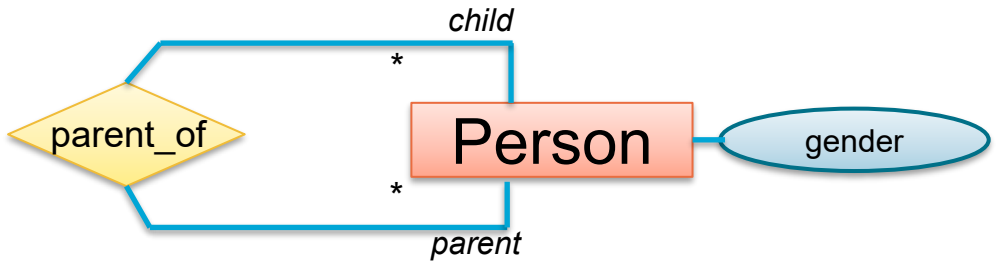


  - Good?

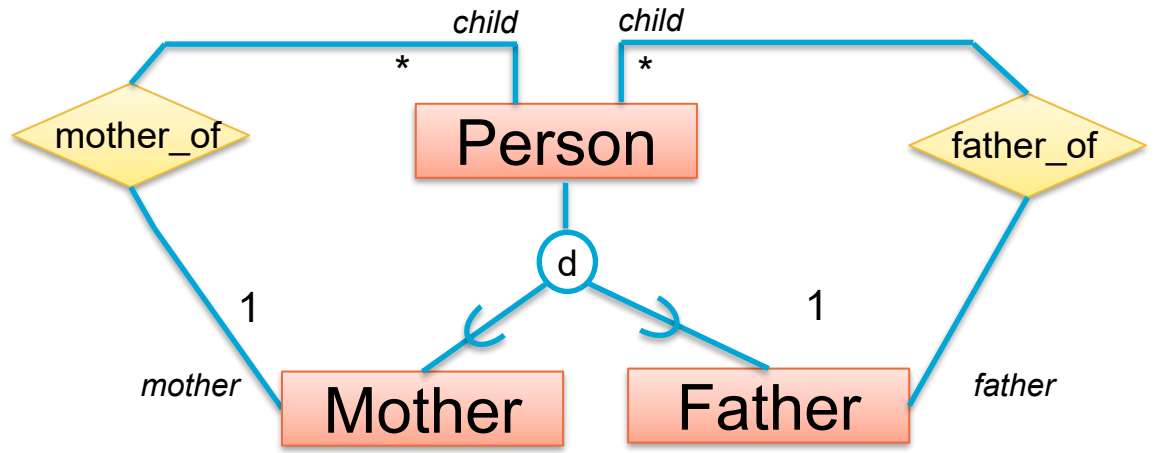


Common Mistakes in Modelling

- Good?

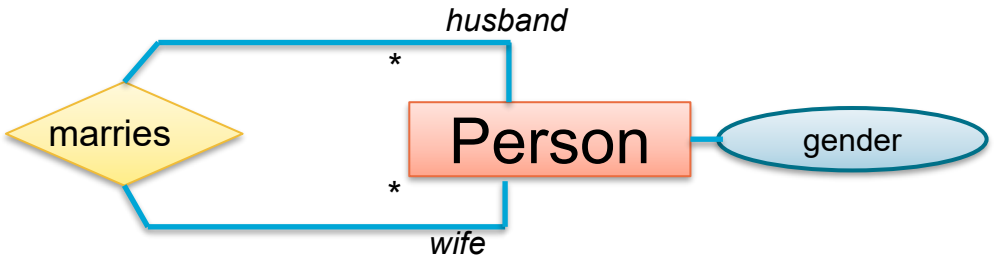


- Good?



Common Problem

- Schema might or might not mean what you think it means
- In this model:



  - You can have group marriages
  - You can have any gender composition in a marriage
    - Labels are just labels, they don't really mean anything
  - YOU CAN MARRY YOURSELF!

Common Mistakes in Modelling

- Mistake: **Primary Key does not make sense**
  - Intuition: **An attribute / set of attributes which uniquely identify an entity**
  - Additional soft constraints
    - Should feel “natural”
    - Should be minimal
    - Should be easy to handle
  - Example: Modeling a book in a bookstore
    - Good: Book(name, author, year, <u>isbn</u>, summary, price)
    - Less good: Book(<u>name, author, year</u>, isbn, summary, price)
      - More natural, but more complex. Only valid if it is guaranteed that a given author writes only a single book with the same name in a year. Depends on task if this makes sense .
    - Not good or even invalid:
      - Book(name, author, year, isbn, <u>summary</u>, price)
      - Book(name, author, year, isbn, summary, <u>price</u>)
  - **Weak Entities** always have composite key
    - One component is the primary key of the strong entity, the second component is with the weak entity and is only unique within the set of weak entities belonging to the same strong one