Information Management
Data Modelling: ER Diagrams

CS1F
LECTURE 3

Dr Craig Macdonald
Craig.Macdonald@glasgow.ac.uk

Database design lifecycle

2

- Requirements analysis
  - User needs; what must database do?
- Conceptual design
  - High-level description; often using E/R model
- Logical design
  - Translate E/R model into (typically) relational schema
- Schema refinement
  - Check schema for redundancies and anomalies
- Physical design/tuning
  - Consider typical workloads, and further optimise

Database Design

- Creating a database involves:
  - **(1) Capturing user requirements**
  - **(2) Representing them in a *model***
  - **(3) Converting model into a *schema***
  - **(4) Implementation on DBMS**
- We will use **Entity Relationship Models**

Talk to all stakeholders: What data do they need stored?

A *conceptual* model, i.e.: What are the concepts? What are the THINGS? What data do we store for each THING?

A *logical* design for the tables in the DBMS, built from the conceptual model

THE ENTITY-RELATIONSHIP MODEL

4

What is an ER Model?

5

- Entity Relationship Model: A **conceptual data model**
  - later mapped to a logical data model or **schema** (i.e. definitions of TABLES)
  - this in turn is mapped to a physical model by the DBMS
- Usually described using ***Entity-Relationship Diagrams***
  - Describes type of information to be stored in a database
  - Provides a pictorial overview and classifications of used terms and their relationships
- The most common method for modelling of a DB

The Entity-Relationship Model

6

- Data in an ER Model is described in terms of three key concepts:
  - Entities
  - Attributes
  - Relationships

Entities

7

- An **entity** is a uniquely identifiable object in the real world about which we wish to store data
  - For example: The Bank of Scotland, The University of Aberdeen, Tony Blair, Celtic Football Club, BBC, my car…….
- A thing which is recognised as being capable of an *independent existence* and which can be ***uniquely identified***

Entities

8

- Entities are grouped together into ‘categories’ called **entity types** or **entity sets**
  - *Employee*, *Department*, *Project*
- An **entity** is an instance of a given entity-type
- There are usually many instances of an entity-type

Entity Types

9

Entity types can be thought of as (common) nouns

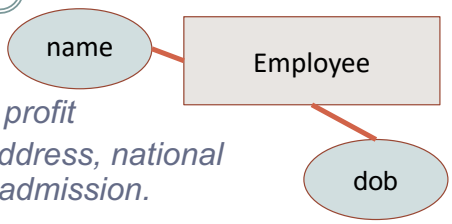
- Can be a **physical object** such as a house or a car
- Can be an **event** such as a house sale or a car service
- Can be a **concept** such as a customer transaction

- Proper nouns indicate instances of entities
  - Joe Bloggs is a Customer

Attributes

10

Attributes are properties that describe an entity (type)



- BT: *name, address, annual profit*
- Patient John Smith: *name, address, national insurance number, date of admission.*

It is expected that all instances of a given entity type will have the same attributes

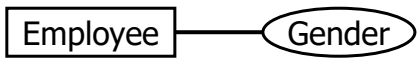
- an **entity type** defines a set of entities that have the same attributes
- i.e. We record same details for employees Jane Black and Gregory White

Attributes are drawn as ovals, and attached to the boxes representing entity types with lines

Attributes: **Simple** (atomic) vs composite

11

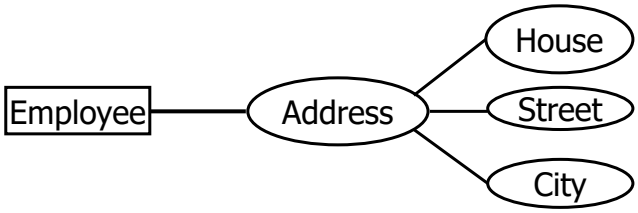
- Simple
  - indivisible value
  - age, gender



Attributes: Simple (atomic) vs **composite**

12

- Composite
  - composed of a set of component values
  - address, date of birth



Other kinds of attribute

13

- Single-valued vs multi-valued
  - multi-valued stores a set of values
  - Indicated by double-lined attribute oval
- Examples:

  locations for a department; hobbies for a person

Primary Key

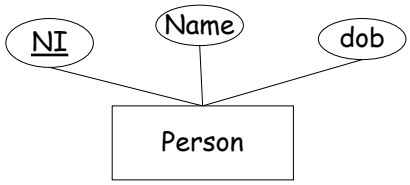
15

- An Entity type will usually have **key attribute(s)**:
  - one (or possibly more) of the attributes which are unique for all entity instances
  - for example
    - A book's ISBN
    - A date (composite attribute M/D/Y)

Key attributes

16

- The primary **key attributes** of an entity type is an attribute whose values are distinct for each entity
- We underline key attributes

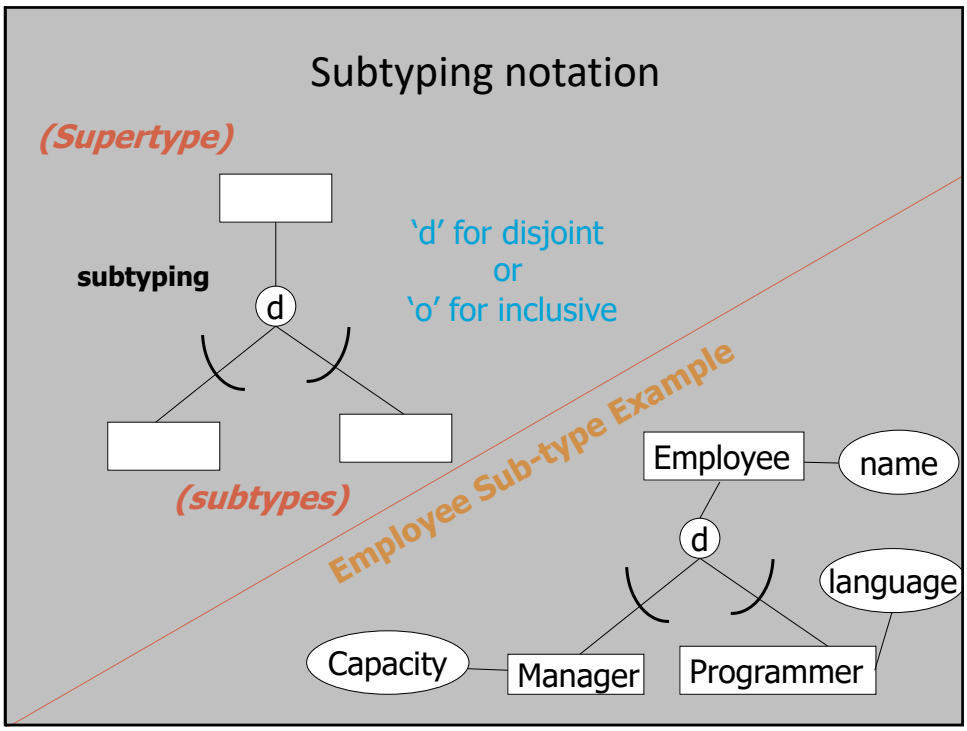


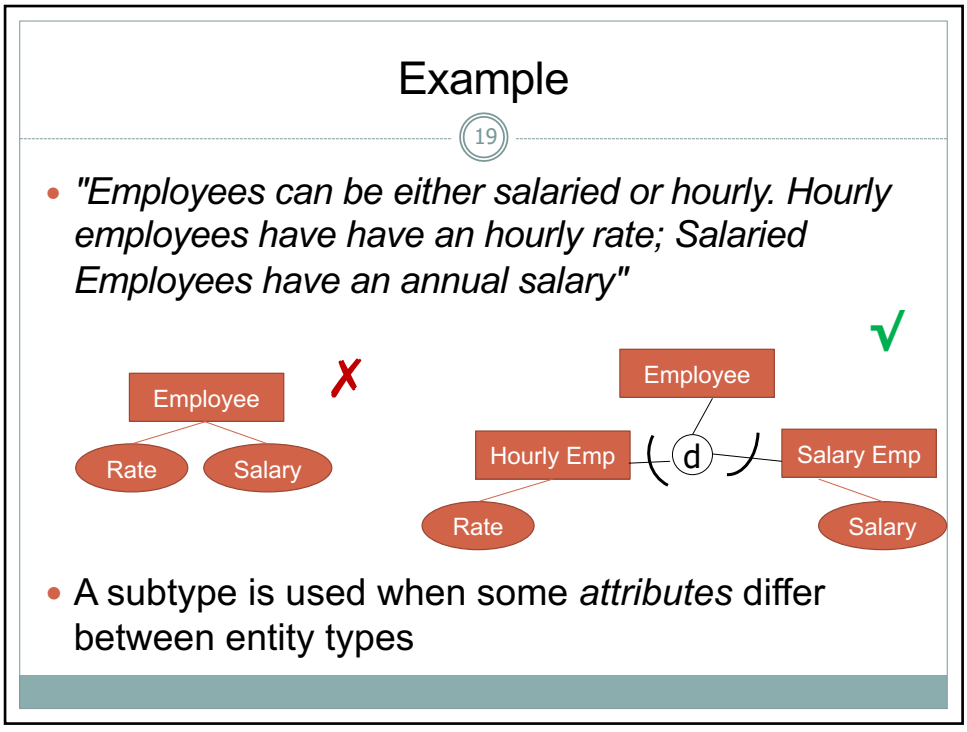
- Sometimes several attributes (a composite attribute) together form a key
  - NB: Such a composite should be **minimal**
  - E.g. The combination of account number AND sort code are unique in UK banking

Subtyping

17

- A subtype is an entity type that *inherits* the properties of its parent type
  - e.g. programmer & manager can be represented as subtypes of employee
- Employee attributes (name, NIN, etc) belong to programmer and manager by virtue of being subtypes of employee
- Subtypes may be
  - disjoint - must belong to exactly one subtype
  - inclusive - may belong to either or both


Subtyping notation
(Supertype)
subtyping
d
'd' for disjoint
or
'o' for inclusive
(subtypes)
Employee Sub-type Example
Employee
name
d
language
Capacity
Manager
Programmer


Example
19
"Employees can be either salaried or hourly. Hourly employees have have an hourly rate; Salaried Employees have an annual salary"
Employee
Rate
Salary
Employee
Hourly Emp
d
Salary Emp
Rate
Salary
A subtype is used when some attributes differ between entity types

Relationships

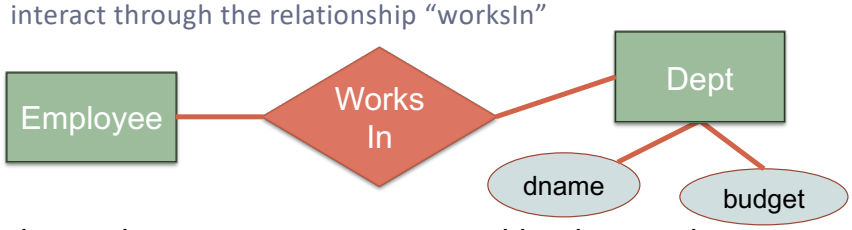
20

- Captures how two or more entity types are related
- Can be thought of as verbs, linking two or more nouns
- Examples:
  - an *owns* relationship between a company and a computer
  - a *supervises* relationship between an employee and a department
  - a *performs* relationship between an artist and a song
  - a *proved* relationship between a mathematician and a theorem

Relationships

21

- **Relationships types** represent the *interaction* between entity types
  - For example the entities in types “employee” and “dept” can interact through the relationship “worksIn”

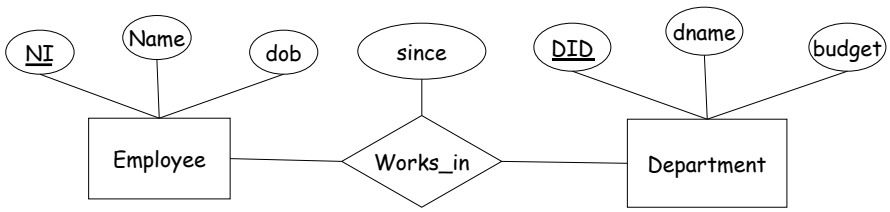


- Relationship types are represented by diamonds
- They connect the participating entity types with straight lines

Relationship attributes

22

- Relationships can also have **attributes**
  - NB: A relationship must be uniquely determined by the participating entities, without reference to the relationship attributes



  - E.g. Craig works_In Computing Science, since 2009
  - E.g. John works_In Computing Science, since 2009

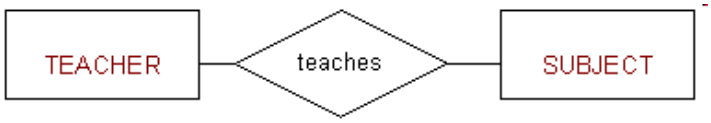
Relationship Degrees

23

- The ***Degree*** of a relationship is the number of entity types participating
  - Binary relationships
    - 2 participating entity types
    - Employee works for Department
  - N-ary (e.g. Ternary) relationships
    - >=3 participating entity types
    - a Manager manages a Project in a Department

Binary Relationships

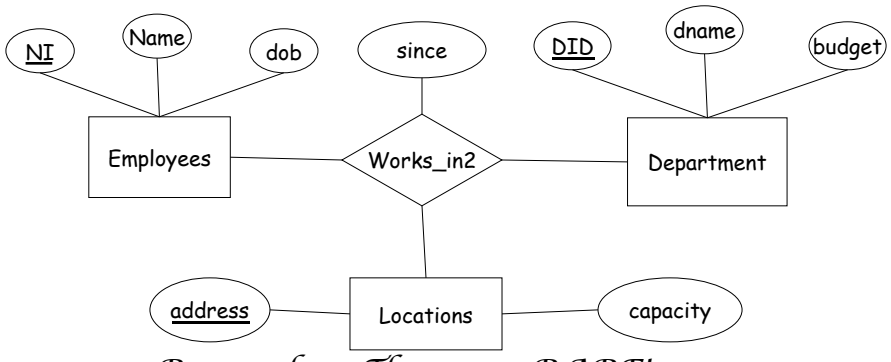
- 2 participating entity types
  - Teacher teaches Subject



N-ary relationships

25

- Although relatively rare, n-ary relationships can exist
  - e.g. ternary (degree 3):



Remember: These are RARE!

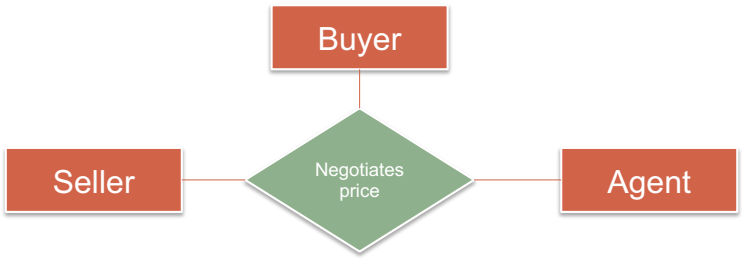
Example: Ternary Relationships

26

3 participating entity types

- An agent negotiates the price between a seller and buyer

Example Ternary Relationship



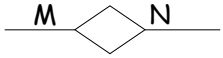
Remember: These are RARE!

Cardinality Constraints on Relationship Types

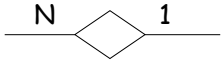
27

- For example:
  - An employee can work in many departments; a department can have many employees
  - In contrast, each department has at most one manager
- The cardinality specifies the number of entity instances that can participate from each side of the relationship of a binary relationship
  - One to one (1:1)
  - One to many (1:N)
  - Many to Many (N:M)

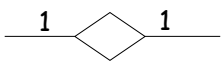
M:N



N:1



1:1



Note: Sometimes this is denoted using different arrowheads

Cardinality – 1:1

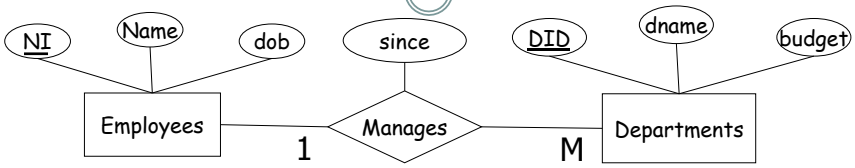
28

- One-to-one (1-1)

- Each manager manages ONLY one project
- Each project is managed by ONLY one manager

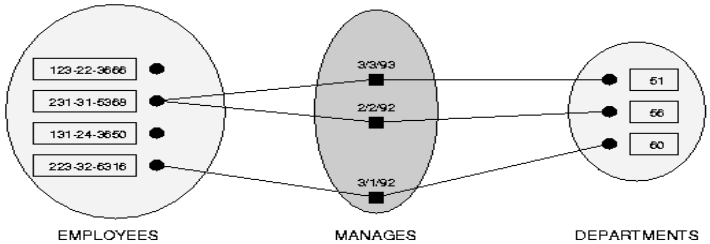
Cardinality – 1:N

29



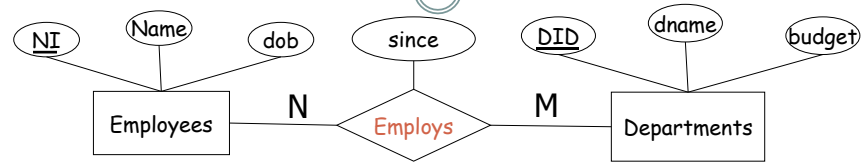
1:N --- One to many

A department cannot have more than one manager
(but it may be that an individual manages multiple departments)



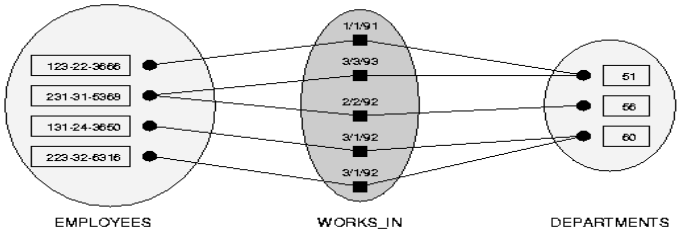
Cardinality – N:M

30



N:M -- Many to many

Departments may employ more than one person at a time, and an individual person may be employed by more than one department

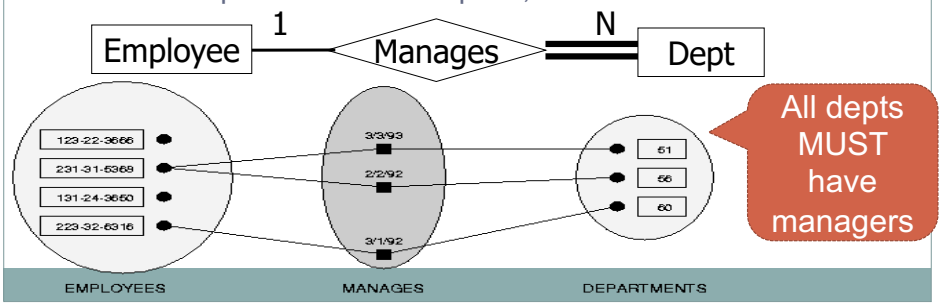


Participation Constraints on Relationships

31

Every department must have a manager

- A double line indicates a *participation constraint* - totality
  - all entities in the entity set must participate in *at least one* relationship in the relationship set;



Participation Constraints on Relationships

32

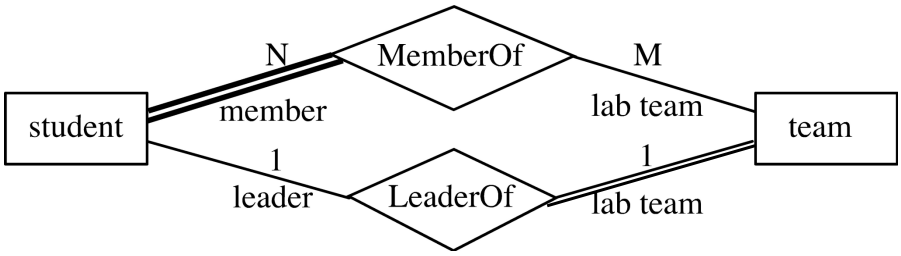
Every department must have a manager

- A double line indicates a *participation constraint* - totality
  - all entities in the entity set must participate in *at least one* relationship in the relationship set;

Cardinality + Participation Constraints = Structural Constraints

Total Participation

33



Every student must be a member of a team
Every team needs a leader

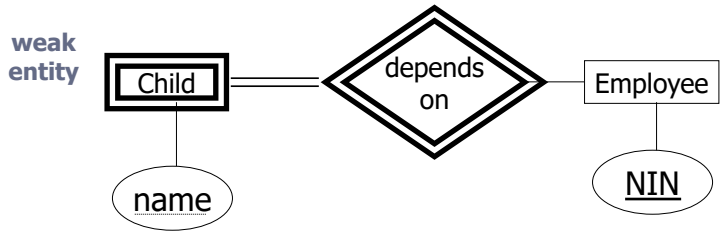
A double line indicates the total participation constraint in an ER model

Note - the participation of *student* in *LeaderOf* is **partial**, because a student *might* be a team leader

Weak Entity Types

34

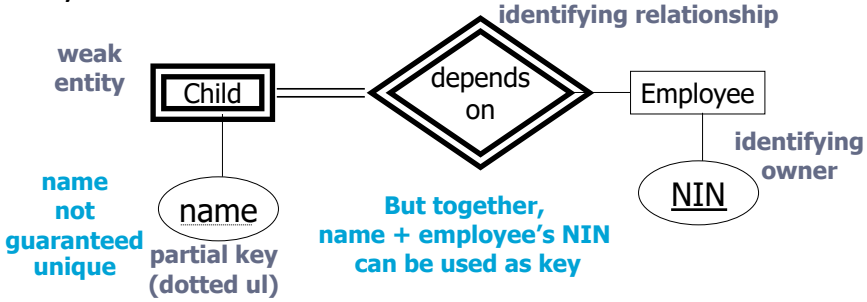
- Do not have primary key (attributes) of their own
- Depend on other entities to guarantee uniqueness



Weak Entity Types

35

- Depend on other entities to guarantee uniqueness
- Do not have sufficient attribute(s) to form a primary key of their own

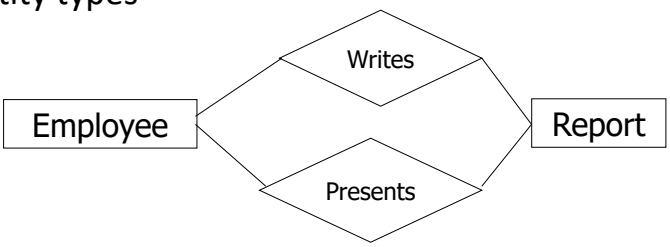


- A weak entity type must have **total participation** in this identifying relationship

More on relationships - 1

36

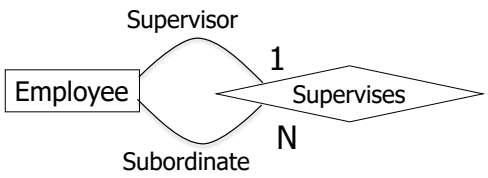
- There may be more than one relationship between entity types



More on relationships - 2

37

- An entity type may be in a relationship with itself
  - this is a recursive relationship

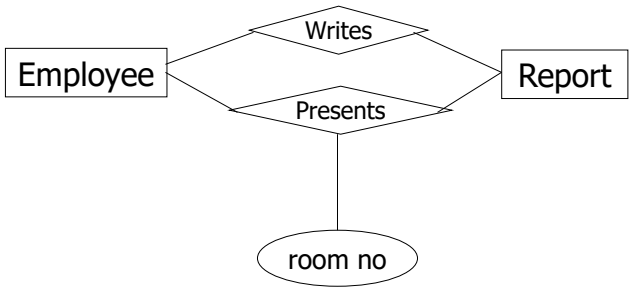


- We *name* the 'roles' of each side of the relationship

More on relationships - 3

38

Recall: relationships may themselves have attributes



From written Scenario to an ER Model

39

- Identify the **Entities**, their **Attributes**, and all **Relationships** involved in any given scenario
- Represent this in an Entity-Relationship Diagram
- ER Diagram (and model) can then be used to implement the actual relationship tables in the database itself (we will do this in the lab in week 3)

Constructing an ER diagram

40

1. Identify the entity types (in boxes)
2. Identify each entity types' properties
3. Decide which properties are attributes (connected to entity in oval)
4. Decide which attributes could be keys
5. Select primary key (underlined attribute)
6. Determine which properties infer relationships (labelled diamond between the participating entities)
7. Decide on the cardinality and participation of the relationship (numbers at entities involved in relationship; single line Vs double line at entity)

THE 'COMPANY' SCENARIO

41

An Example Scenario

42

A company has *a* set of departments. Each department has a name, number, manager and possibly several locations. The manager is an employee and started managing the department on a given date. A department controls several projects, each with a name, number and location

Each employee has a name, address, salary, supervisor, department, sex, date of birth and national insurance number. An employee may work on many projects, not all in their own department, and works X hours on each of these projects. Each employee has a set of dependants, each with a name, date-of-birth, sex and familial relationship to the employee.

(1) IDENTIFY ENTITIES IN THE 'COMPANY' SCENARIO

The Example Scenario

44

A company has *a* set of **departments**. Each department has a name, number, manager and possibly several locations. The manager is an employee and started managing the department on a given date. A department controls several **projects**, each with a name, number and location

Each **employee** has a name, address, salary, supervisor, department, sex, date of birth and national insurance number. An employee may work on many projects, not all in their own department, and works X hours on each of these projects. Each employee has a set of **dependants**, each with a name, date-of-birth, sex and familial relationship to the employee.

Entities in the Company Scenario

45

Departments, Employees, Projects, Dependants

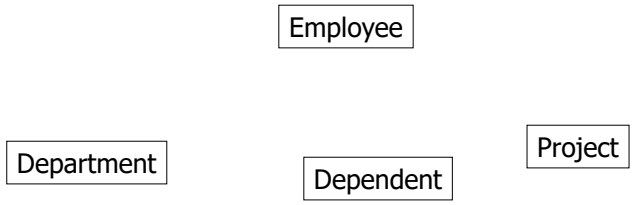
Notes

- Company is **not** an entity type - it is the whole database
- Some things are relationships rather than entities themselves
  - Managers ? "The **manager** *is an* **employee** "
  - Supervisors ? "Each **employee** has a **supervisor**"

Entities in Company Scenario

46

- How to represent an entity in an ER diagram



(2) IDENTIFY ATTRIBUTES IN COMPANY SCENARIO

The Example Scenario

48

A company has *a* set of departments. Each department has a **name**, **number**, manager and possibly several **locations**. The manager is an employee and started managing the department on a **given date**. A department controls several projects, each with a **name**, **number** and **location.**

Each employee has a **name**, **address**, **salary**, supervisor, department, **sex**, **date of birth** and **national insurance number**. An employee may work on many projects, not all in their own department, and works X hours on each of these projects. Each employee has a set of dependants, each with a **name**, **date-of-birth**, **sex** and **familial relationship** to the employee.

Attributes in the Company Scenario

49

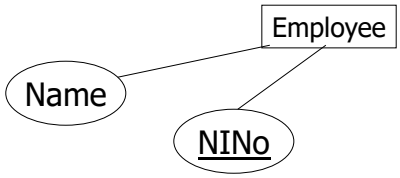
- The attributes of the company database are:
  - Department - name, number, {locations}
  - Employee - National Insurance Number, **name**, **address**, salary, sex, birthdate
  - Project - name, number, location
  - Dependent - **name**, sex, DofB, relationship

Note – again – watch out – don't simply make everything an attribute....some things are relationships, or attributes of relationships – not the entity itself

Attributes in the Company Scenario

50

- How to represent attributes of an entity in an ER diagram:



(3) IDENTIFY RELATIONSHIPS IN COMPANY SCENARIO

The Example Scenario

52

A ***company*** has *a* set of ***departments***. Each department has a name, number, manager and possibly several locations. The manager is an employee and started managing the department on a given date. A ***department*** controls several ***projects***, each with a name, number and location

Each employee has a name, address, salary, supervisor, department, sex, date of birth and national insurance number. An ***employee*** may work on many ***projects***, not all in their own department, and works X hours on each of these projects. Each employee has a set of dependants, each with a name, date-of-birth, sex and familial relationship to the employee.

Relationships in the Company Scenario

53

- A ***company*** *has a* set of **departments**
- A **department** *controls* several **projects**
- An **employee** may *work on* many **projects**, and *works* X hours on each of these **projects**.

Relationships in the Company Scenario

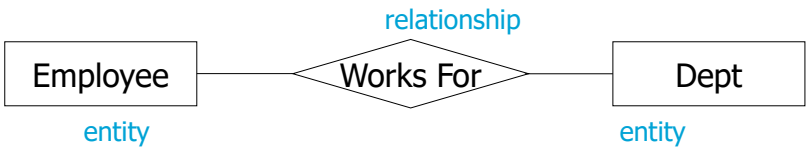
54

Relationships with their own attributes

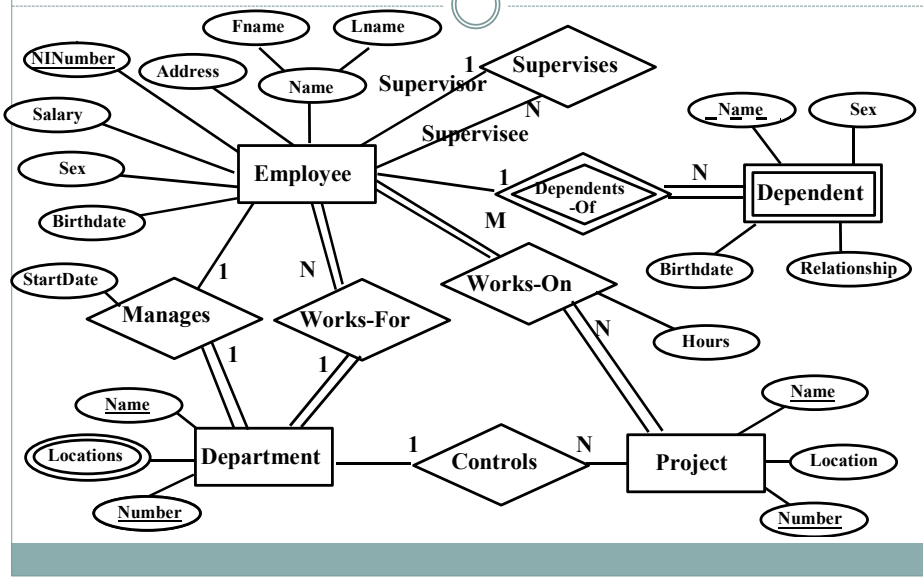
- Each **employee** *has a* set of **dependants**, each with a **name, date-of-birth, sex and familial relationship** to the **employee**.
- The **manager** *is an* **employee** and *started managing* the **department** on a given date

Representing Relationships in an ER Diagram

55



ER Diagram for the Company Database



ER Diagram Notation

- strong entity type
- weak entity type
- attribute
- relationship
- multi-valued attribute
- Identifying relationship
- name — key attribute
- 1 — N — relationship cardinality
- composite attribute
- total participation
- partial participation

Constructing an ER diagram

59

1. Identify the entity types (in boxes)
2. Identify each entity types' properties
3. Decide which properties are attributes (connected to entity in oval)
4. Decide which attributes could be keys
5. Select primary key (underlined attribute)
6. Determine which properties infer relationships (labelled diamond between the participating entities)
7. Decide on the cardinality and participation of the relationship (numbers at entities involved in relationship; single line Vs double line at entity)

Essential Reading

60

After this lecture:

- Garcia-Molina, Chapter 4
  - Sections 1 -1.5 ✔
    Sections 1.5-1.9, 1.11

- OR

- From Franconi's notes on ER modelling:
  - http://www.inf.unibz.it/~franconi/teaching/2000/ct481/er-modelling/

Reminder

- Tutorials start this week
- You will be working on an E/R diagram