Entity Relationship Modelling

P.J. McBrien

Imperial College London

Designing a Relational Database Schema

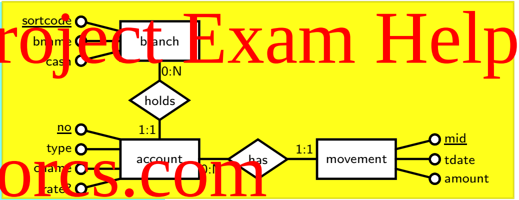
How do you design a relational database schema for a particular UoD?

1. Need some way to model the semantics of the UoD as a conceptual schema
   - ER (many variants exist)
   - UML class diagrams
2. Need to map the ER/UML schema into a relational schema
3. Need to ensure that the relational schema is a good design
   - Normalisation

Semantic Modelling: ER Schemas


```
CREATE TABLE branch
(       sortcode INTEGER NOT NULL,
        bname VARCHAR(20) NOT NULL,
        cash DECIMAL(10,2) NOT NULL,
        CONSTRAINT branch_pk PRIMARY KEY (sortcode)
)
CREATE TABLE account
(       no INTEGER NOT NULL,
        type CHAR(8) NOT NULL,
        cname VARCHAR(20) NOT NULL,
        rate DECIMAL(4,2) NULL,
        sortcode INTEGER NOT NULL,
        CONSTRAINT account_pk PRIMARY KEY (no),
        CONSTRAINT account_fk FOREIGN KEY (sortcode) REFERENCES branch
)

CREATE INDEX account_type ON account (type)
CREATE TABLE movement
(       mid INTEGER NOT NULL,
        no INTEGER NOT NULL,
        amount DECIMAL(10,2) NOT NULL,
        tdate DATETIME NOT NULL,
        CONSTRAINT movement_pk PRIMARY KEY (mid),
        CONSTRAINT movement_fk FOREIGN KEY (no) REFERENCES account
)
```



Core $\mathcal{ER}$: Entities and Relationships

Entities

$\mathcal{E}$ An entity E represents a set of objects which conceptually are the same type of thing.

- **nouns** $\to$ entity set
- proper nouns imply instances, which are not entity sets.

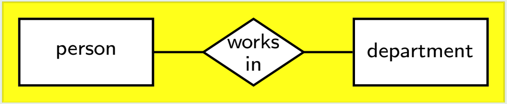
Relationships

$\mathcal{R}$ A relationship R represents a set of tuples of objects where each tuple is some type of conceptual association between entities E_1, E_2

- **verbs** $\to$ relationship
- $R \subseteq \{\langle e_1, e_2 \rangle \mid e_1 \in E_1 \land e_2 \in E_2\}$

Identifying entities and relationships

In News Ltd, each person works in exactly one department; there are no restrictions on the number of persons a department may employ.



Core $\mathcal{ER}^{\mathcal{KMO}}$: Attributes of Entities

Attributes $\mathcal{ER}^{\mathcal{M}}$ $\mathcal{ER}^{\mathcal{O}}$ and $\mathcal{ER}^{\mathcal{K}}$

$\mathcal{M}$ A mandatory attribute $E.A$ is a function that maps from entity set E to value set V.

1. $E.A \subseteq \{\langle e, v\rangle \mid e \in E \wedge v \in V\}$
2. unique: $\langle e, v_1\rangle \in E.A \wedge \langle e, v_2\rangle \in E.A \rightarrow v_1 = v_2$
3. mandatory: $E = \{e \mid \langle e, v\rangle \in E.A\}$

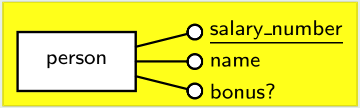
adjective, **adjective noun** $\rightarrow$ attribute

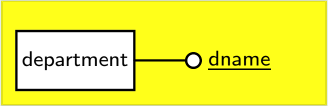
$\mathcal{O}$ an **optional attribute** removes property (3)

$\mathcal{K}$ certain attribute(s) $E.A_1 \ldots E.A_n$ of E are denoted **key attributes** such that $E = \{\langle v_1, \ldots, v_n\rangle | \langle e, v\rangle \in E.A_1 \wedge \ldots \wedge \langle e, v_n\rangle \in E.A_n\}$

Identifying attributes

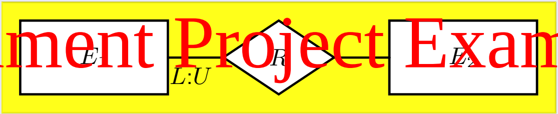
We record the name of each person working in the department; and identify them by their salary number. Optionally they might have a bonus figure recorded. Departments are identified by their name.





$\mathcal{ER}^{\mathcal{L}}$: Look-Here Cardinality Constraints

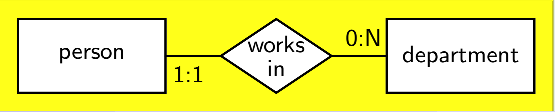
$\mathcal{ER}^{\mathcal{L}}$



- An upper bound cardinality constraint U states that each instance of E_1 may appear at most U times in R. An upper bound of N indicates no limit.
- Additionally with $\mathcal{ER}^{\mathcal{O}}$: a lower bound cardinality constraint L states that each instance of E_1 must appear at least L times in R

Adding look-here cardinality constraints in $\mathcal{ER}^{\mathcal{LO}}$

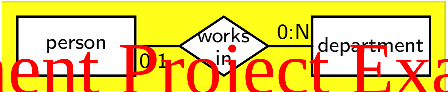
Each person works in exactly one department; there are no restrictions on the number of persons a department may employ.



Quiz 1: Extent of Relationships

person = {‘Peter’, ‘Jane’, ‘Mary’}

dept = {‘CS’, ‘Maths’}



Which is not a possible extent of works_in?

A

works_in={⟨‘Peter’,‘Maths’⟩, ⟨‘Peter’,‘CS’⟩, ⟨‘Mary’,‘Maths’⟩, ⟨‘Jane’,‘Maths’⟩}

B

works_in={⟨‘Peter’,‘Maths’⟩, ⟨‘Mary’,‘Maths’⟩, ⟨‘Jane’,‘Maths’⟩}

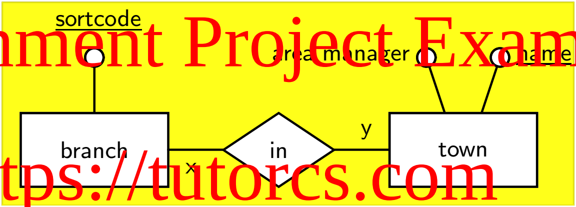
C

works_in={⟨‘Peter’,‘CS’⟩, ⟨‘Mary’,‘Maths’⟩, ⟨‘Jane’,‘Maths’⟩}

D

works_in={⟨‘Peter’,‘CS’⟩, ⟨‘Jane’,‘Maths’⟩}

Quiz 2: Cardinality Constraints on Relationships



Branches based in towns are all assigned to an area manager for that town; and area managers are only assigned to towns that have branches

What should be the cardinality constraints of in?

A
x = 1:1, y = 0:N

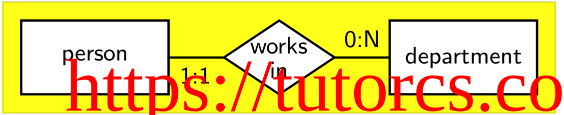
B
x = 0:1, y = 0:N

C
x = 0:N, y = 1:N

D
x = 0:1, y = 1:N

ER^C: Look-Across Cardinality Constraints

- This course uses **look-here** cardinality constraints: state the number of occurrences of the entity next to the constraint



- Other variants of ER modelling use **look-across** cardinality constraints

- For binary relationships, ER^C and ER^L are equally expressive.

$\mathcal{ER}^{\mathcal{S}}$: Subset/isa hierarchies

$\mathcal{ER}^{\mathcal{S}}$

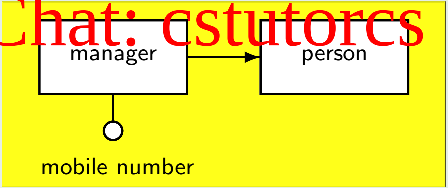
S: If it is found that the instances of one entity E_s are a subset of a another entity E, we may add a **subset** constraint.

$E_s \subseteq E$

- **specialisation of nouns** $\rightarrow$ subset

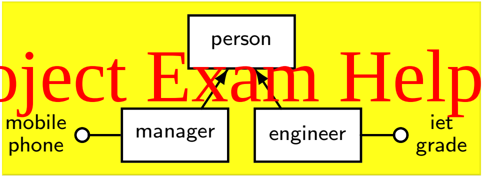
Identifying subsets with $\mathcal{ER}^{\mathcal{S}}$

Some employees are ranked as managers, and receive a mobile phone.



Quiz 3: Extent of subset and superset entities

manager={'Jane','Mary'}



Which is not a possible extent of person and engineer?

A

person={'Peter','Jane','Mary'}
engineer={'Jane','Mary'}

B

person={'Peter','Jane','Mary','John'}
engineer={}

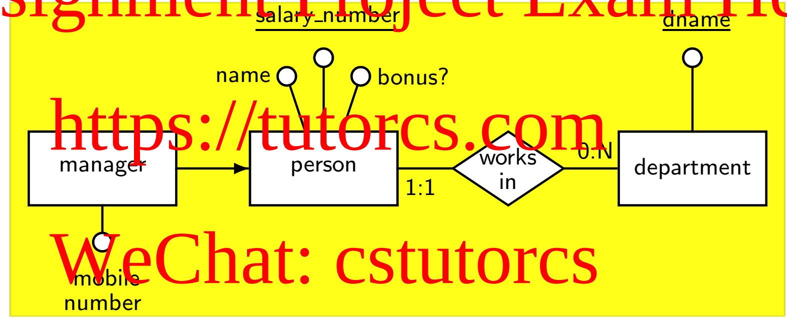
C

person={'Peter','Jane','Mary'}
engineer={'John'}

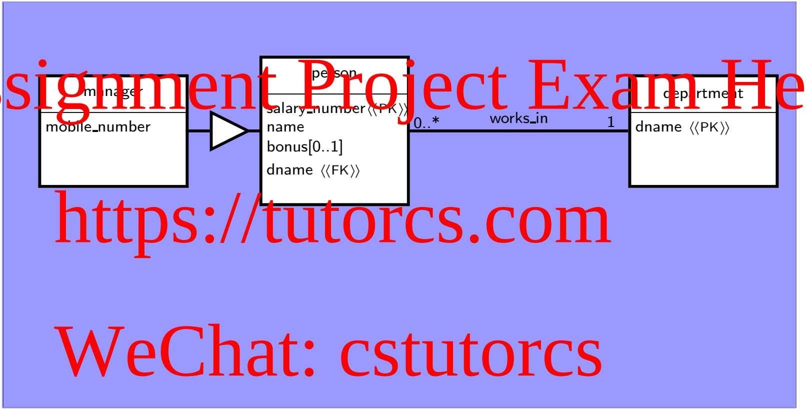
D

person={'Peter','Jane','Mary','John'}
engineer={'Peter','John'}

Combining Fragments



Using UML Class Diagrams as ER Models



How to Use UML Class Diagrams as an ER Schema

Use UML stereotypes to denote at least primary key information

Various approaches exist

ER Modelling Constructs $\mathcal{CKLMOS}$

Construct	Description
$\mathcal{C}$	Look-across cardinality constraints
$\mathcal{L}$	Look-here cardinality constraints
$\mathcal{K}$	Key attributes
$\mathcal{M}$	Mandatory attributes
$\mathcal{O}$	Optional attributes
$\mathcal{S}$	Isa hierarchy between entities

A particular ER Modelling language normally chooses between $\mathcal{C}$ or $\mathcal{L}$

Worksheet: ER Modelling

Draw an $\mathcal{ER}^{\mathcal{KCMOS}}$ schema to describe the following domain

> *The payroll system for* BIG Inc *records the salaries, status, joining date, name, and payroll number for all of the corporation's 30,000 employees. Each employee works for one division, and each division has an account number for paying its staff. We identify divisions by their name, and record the address where the division's HQ is located.*
> *For employees sent abroad by* BIG Inc*, we record the address, country and telephone number of the foreign tax office that will handle the employee. It is assumed that each country has one central tax office that we have to deal with. All other employees have their tax affairs dealt with by the Inland Revenue.*

Worksheet: ER Modelling

Draw an $\mathcal{ER}^{\mathcal{KCMOS}}$ schema to describe the following domain

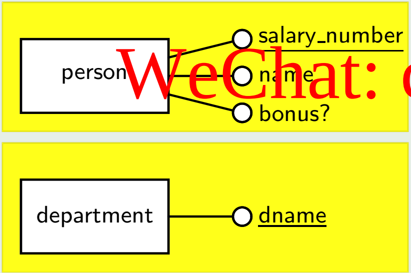
> *The payroll system for* BIG Inc *records the salaries, status, joining date, name, and payroll number for all of the corporation's 30,000 employees. Each employee works for one division, and each division has an account number for paying its staff. We identify divisions by their name, and record the address where the division's HQ is located.*
>
> *For employees sent abroad by* BIG Inc, *we record the address, country and telephone number of the foreign tax office that will handle the employee. It is assumed that each country has one central tax office that we have to deal with. All other employees have their tax affairs dealt with by the Inland Revenue.*

Mapping $\mathcal{ER}^{\mathcal{KLMOS}}$ to a relational model: entities and attributes

Taking a **table per type** (**TPT**) approach, there is a simple mapping of entities and attributes to tables and columns:

1. Each entity E maps to a table R_E
2. Each attribute A maps to a column C_A of R_E
3. If A is an optional attribute, then C_A is nullable, otherwise C_A is not nullable
4. If $\vec{K}$ are key attribute(s), then $\vec{C_K}$ are a key of R_E

Tables generated from entities



person(salary_number,name,bonus?)
department(dname)

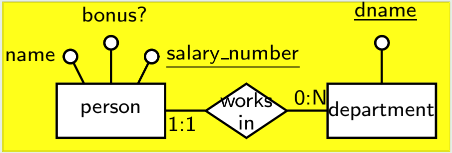
Mapping $\mathcal{ER}^{\mathcal{KLMOS}}$ to a relational model: relationships

Taking a **table per type** (**TPT**) approach, for each relationship R between E_1, E_2, entities E_1, E_2 map to R_1, R_2 as before, and

1. If R is a many-many relationship then it maps to
   1. a table $R\_R_1\_R_2(\vec{K_1}, \vec{K_2})$
   2. a foreign key $R\_R_1\_R_2(\vec{K_1}) \overset{fk}{\Rightarrow} R_1(\vec{K_1})$
   3. a foreign key $R\_R_1\_R_2(\vec{K_2}) \overset{fk}{\Rightarrow} R_2(\vec{K_2})$

2. If R is a one-many relationship then it maps to
   1. a column $\vec{K_2}$ in R_1
   2. a foreign key $R_1(\vec{K_2}) \overset{fk}{\Rightarrow} R_2(\vec{K_2})$
   3. if the participation of E_1 in R is optional, then $\vec{K_2}$ is an optional column of R_1

Tables generated from relationships



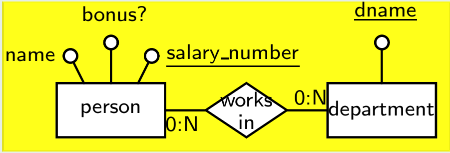
person(<u>salary_number</u>,name,bonus?,dname)
department(<u>dname</u>)
person(dname) $\overset{fk}{\Rightarrow}$ department(dname)

Mapping $\mathcal{ER}^{\mathcal{KLMOS}}$ to a relational model: relationships

Taking a **table per type** (**TPT**) approach, for each relationship R between E_1, E_2, entities E_1, E_2 map to R_1, R_2 as before, and

1. If R is a many-many relationship then it maps to
   1. a table $R\_R_1\_R_2(\vec{K_1}, \vec{K_2})$
   2. a foreign key $R\_R_1\_R_2(\vec{K_1}) \stackrel{fk}{\Rightarrow} R_1(\vec{K_1})$
   3. a foreign key $R\_R_1\_R_2(\vec{K_2}) \stackrel{fk}{\Rightarrow} R_2(\vec{K_2})$
2. If R is a one-many relationship then it maps to
   1. a column $\vec{K_2}$ in R_1
   2. a foreign key $R_1(\vec{K_2}) \stackrel{fk}{\Rightarrow} R_2(\vec{K_2})$
   3. if the participation of E_1 in R is optional, then $\vec{K_2}$ is an optional column of R_1

Tables generated from relationships



person(<u>salary_number</u>,name,bonus?)
department(<u>dname</u>)

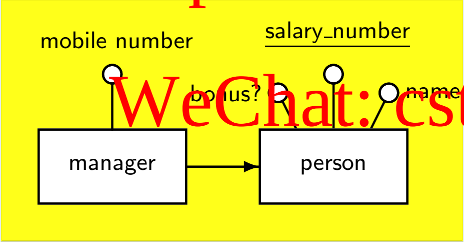
works_in(<u>salary_number</u>,<u>dname</u>)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

Mapping $\mathcal{ER}^{\mathcal{KLMOS}}$ to a relational model: subsets

Taking a **table per type** (**TPT**) approach, for each subset E_s of E, entities E_s, E map to tables R_s, R as before and:

1. a key $\vec{K}$ in R_s (where $\vec{K}$ is the key of R)
2. a foreign key $R_s(\vec{K}) \overset{fk}{\Rightarrow} R(\vec{K})$

Tables generated from subsets



person(<u>salary_number</u>,name,bonus?)

manager(<u>salary_number</u>,mobile_phone)

manager(salary_number) $\overset{fk}{\Rightarrow}$ person(salary_number)

Worksheet: Mapping $\mathcal{ER}^{\mathcal{KLMOS}}$ to a relational model

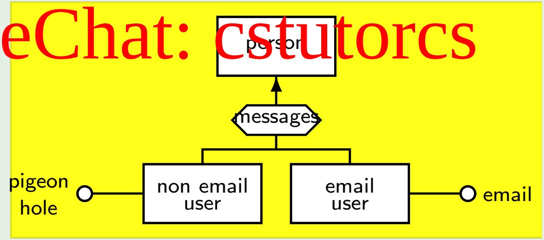
Take your $\mathcal{ER}^{\mathcal{KLMOS}}$ schema in the worksheet, and map it into a relational schema.

$\mathcal{ER}^{\mathcal{D}}$: Disjointness and Generalisation Hierarchies

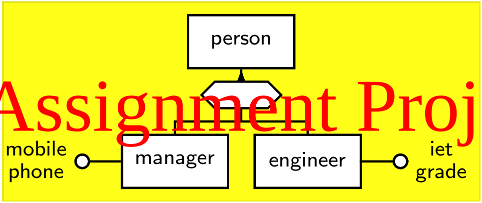
- In $\mathcal{ER}^{\mathcal{D}}$: the disjointness of entities $E_1 \ldots E_n$ may be specified, enforcing that $\forall x, y . x \neq y \rightarrow E_x \cap E_y = \emptyset$
- The notion of **generalisation hierarchies** combines the use of disjointness and subset.
- **disjoint specialisation of nouns** $\rightarrow$ generalisation

Identifying generalisation hierarchies in $\mathcal{ER}^{\mathcal{SD}}$

Employees may also be divided, according to how they like to receive messages, into email users and non-email users. The former must have a email address recorded, the later must have a pigeon hole number recorded.



Quiz 4: Extent of generalisation entities



Which is not a possible extent of the entities?

A

person={'Peter','Jane','Mary','John'}
engineer={'Peter','John'}
manager={'Jane','Mary'}

B

person={'Peter','Jane','Mary','John'}
engineer={}
manager={'Jane','Mary'}

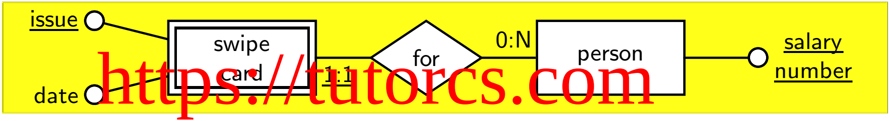
C

person={'Peter','Jane','Mary','John'}
engineer={'John'}
manager={'Jane','Mary'}

D

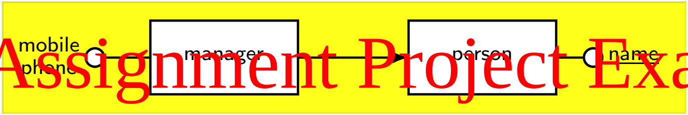
person={'Peter','Jane','Mary','John'}
engineer={'Peter','John','Mary'}
manager={'Jane','Mary'}

$\mathcal{ER}^{\mathcal{W}}$: Weak entities



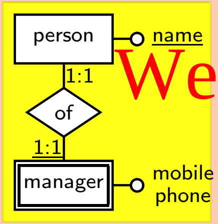
- If we allow the participation of an entity in a relationship to be part of the entity key, we have a **weak entity**

Quiz 5: Subsets and weak entities

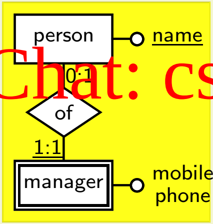


Which of the following is equivalent to the schema above?

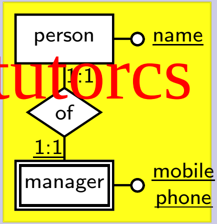
A



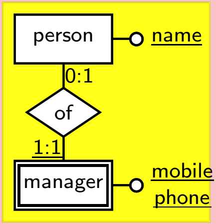
B



C



D

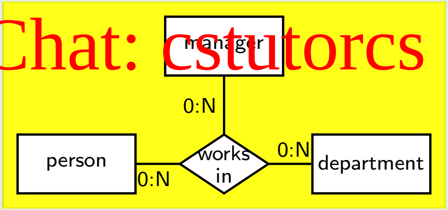


$\mathcal{ER}^{\mathcal{H}}$: Allowing an n-ary relationship

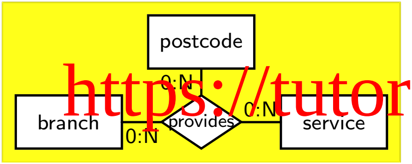
- In graph theory, an edge connecting more that two nodes is called a **hyper-edge**.
- In $\mathcal{ER}^{\mathcal{H}}$: allow n-ary relationships between entities, rather than just binary
- An n-ary relationship is equivalent to a weak entity with n binary relationships

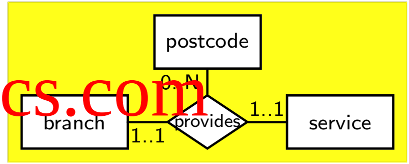
Identifying an n-ary relationship

A person may work in multiple departments, and for each department the person works in, the person will be assigned a manager



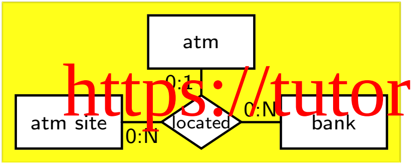
Ternary Relationships: Inability to Express Constraints in $\mathcal{ER}^{\mathcal{LH}}$

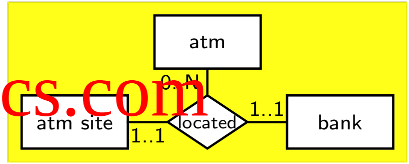




each **branch** *provides only one type of* **service** *in any* **postcode** *area, and each* **service** *is only provided one* **branch** *in any* **postcode** *area*

Ternary Relationships: Inability to Express Constraints in $\mathcal{ER}^{\mathcal{CH}}$





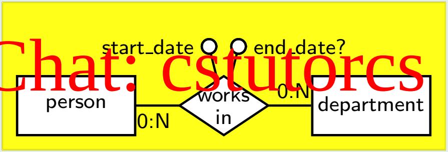
an **atm** *machine from a leasing company may be assigned to a particular* **bank** *at a particular* **site** *but* **banks** *do not have exclusive use of a* **site**

$\mathcal{ER}^{\mathcal{A}}$: Allowing attributes on relationships

- Use when there are values to be associated with the relationship between entities

Identifying an attribute of a relationship

We record the **start_date** *when a person joined a department, and when the person leaves, record the* **end_date** *they left the department. We keep a history of all departments the person worked in.*

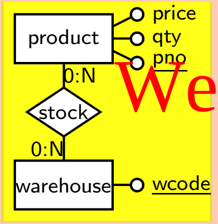


Quiz 6: Appropriate use of attributes on relationships

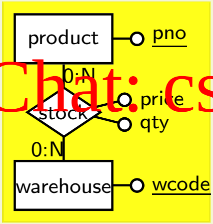
In the stock control system, we identify products by the pno*, and keep our stock in a number of warehouses identified by* wcode*. We record single* price *of each product, and the quantity* qty *of product we keep in each warehouse.*

Which of the following best models the above domain?

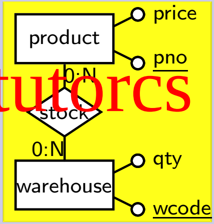
A



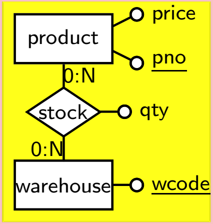
B



C



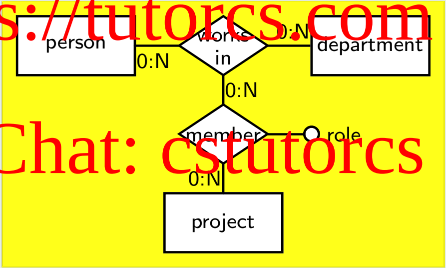
D



$\mathcal{ER}^{\mathcal{N}}$: Allowing nested relationships

Identifying a nested relationship

When a person works in a department, they may work on any number of projects with a certain role. People may take different roles on the project for each department that they work in.

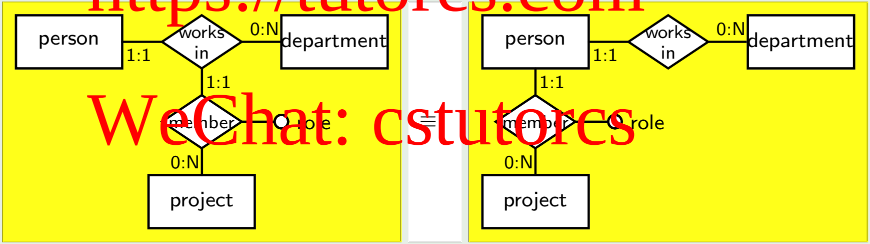


Nested relationship equivalences

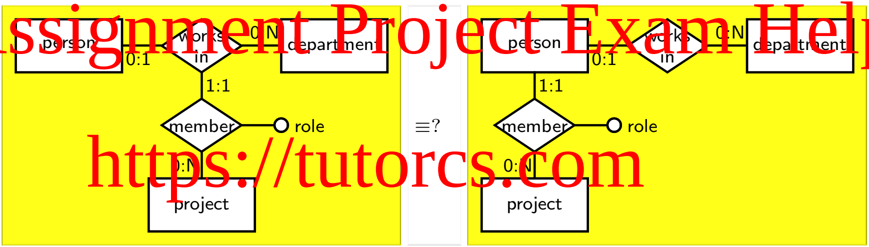
Need for using nested relationships

If a relationship to which a nested edge connects is mandatory and unique with entity E, then the nested relationship can instead connect to E

Equivalent ER Schemas



Quiz 7: Nested relationship equivalences



Are the two ER schemas equivalent?

True

False

$\mathcal{ER}^{\mathcal{V}}$: Multi-valued Attributes

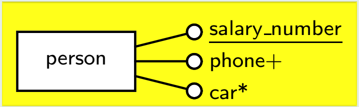
Multi-valued Attributes

- $\mathcal{M}$ A mandatory attribute $E.A$ is a function that maps from entity set E to value set V.
  1. $E.A \subseteq \{\langle e,v\rangle | e \in E \wedge v \in V\}$
  2. unique: $\langle e,v_1\rangle \in E.A \wedge \langle e,v_2\rangle \in E.A \rightarrow v_1 = v_2$
  3. mandatory: $E = \{e \mid \langle e,v\rangle \in E.A\}$

  adjective, adjective noun $\rightarrow$ attribute
- $\mathcal{O}$ an **optional attribute** removes property (3) ?
- $\mathcal{V}$ a **multi-valued attribute** removes property (2) +
- an attribute can be both optional and multi-valued *

Identifying multi-valued attributes

Each person must have at least one home phone number recorded, and may have any number of cars registered as having access to the car park.



EER Modelling Constructs $\mathcal{ADHKLMNOSVW}$

EER

Define **Extended ER (EER)** modelling language as one that supports $\mathcal{KLMOS}$ plus at least one of $\mathcal{ADHNVW}$

Construct	Description
$\mathcal{A}$	Attributes can be placed on relationships
$\mathcal{D}$	Disjointness between sub-classes can be denoted
$\mathcal{C}$	Look-across cardinality constraints
$\mathcal{H}$	hyper-edges (n-ary relationships) allowed
$\mathcal{L}$	Look-here cardinality constraints
$\mathcal{K}$	Key attributes
$\mathcal{M}$	Mandatory attributes
$\mathcal{N}$	Nested relationships
$\mathcal{O}$	Optional attributes
$\mathcal{S}$	Isa hierarchy between entities
$\mathcal{V}$	Multi-valued attributes
$\mathcal{W}$	Weak entities can be identified

Worksheet: Constructing an $\mathcal{ER}^{\mathcal{ADHKLMOSW}}$ Schema

The customer and supplier database of Big Inc will hold all accounts of the company, divided into customer accounts and supplier accounts. All accounts have an account number, and one account manager assigned from the company's staff. Big Inc identifies staff by a **sid**, *and records the staff member's name and room. The account managers have a limit on the number of accounts they can manage. Only certain staff members are permitted to be account managers.*

For customer accounts we need to record a credit limit on the balance of the account, and the telephone number of the accounts department at the customer.

For supplier accounts we need to record which Big Inc products are supplied, and at what price.

Big Inc products are identified by the company standard **part_no** *and all have a description. For some we record the colour. Some products have a record of the components, each component identified by a combination of* **part_no** *and component number, and again each has a description. Some products do not have a supplier.*

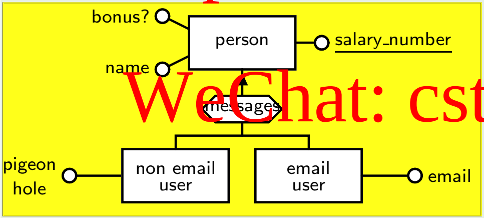
Big Inc has purchased a copy of the Post Office address file, and associates every account to an address from this file. The address data includes street number, street name, town, county and post code, and uses a combination of street number and post code as a key.

Mapping $\mathcal{ER}^{\mathcal{D}}$ to a relational model

Taking a **table per type** (**TPT**) approach, if E is a generalisation of $E_1, \ldots, E_n$, then entities $E_1, \ldots, E_n, E$ map to tables $R_1, \ldots, R_n, R$ as before and:

1. treat each $E_x \in E_1, \ldots, E_n$ as a subset of E
2. no implementation of disjointness using just PKs and FKs

Tables generated from generalisations



person(<u>salary_number</u>,name,bonus?)

non_email_user(<u>salary_number</u>,pigeon_hole)

non_email_user(salary_number) $\overset{fk}{\Rightarrow}$ person(salary_number)

email_user(<u>salary_number</u>,email)

email_user(salary_number) $\overset{fk}{\Rightarrow}$ person(salary_number)

Mapping $\mathcal{ER}^{\mathcal{W}}$ to a relational model

- If E_W is a weak entity that maps to a relation R_W, the foreign key R_K due to the participation in a relationship is also used in the key of R_K

Tables generated from weak entities

person(<u>salary_number</u>,name,bonus?)
swipe_card(<u>salary_number</u>,<u>issue</u>,date)
swipe_card(salary_number) $\overset{fk}{\Rightarrow}$ person(salary_number)

Mapping $\mathcal{ER}^{\mathcal{W}}$ to a relational model

- If E_W is a weak entity that maps to a relation R_W, the foreign key R_K due to the participation in a relationship is also used in the key of R_K

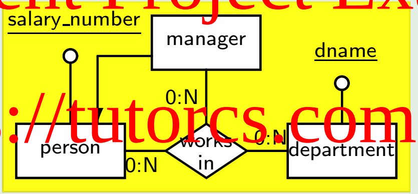
Tables generated from weak entities

person(salary_number,name,bonus?)
swipe_card(salary_number,issue,date)
swipe_card(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)

Mapping $\mathcal{ER}^{\mathcal{H}}$ to a relational model

Rules for binary relationship R between E_1, E_2 generalise to rules for R between $E_1, \ldots, E_n$

Tables generated from n-ary entities



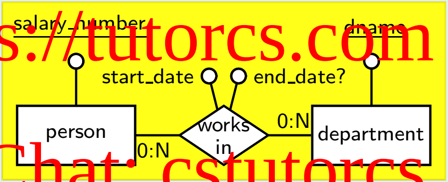
person(salary_number)
manager(salary_number)
manager(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
department(dname)
works_in(person_salary_number,manager_salary_number,dname)
works_in(person_salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(manager_salary_number) $\stackrel{fk}{\Rightarrow}$ manager(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

Mapping $\mathcal{ER}^{\mathcal{A}}$ to a relational model

Attributes on Relationships

Attributes of a relationship go on the same table as that which implements the relationship

Tables generated from attributes of relationships



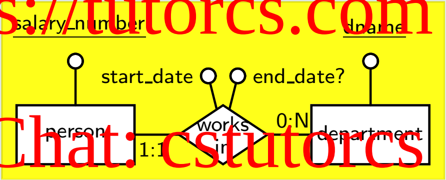
person(salary_number)
department(dname)
works_in(salary_number,dname,start_date,end_date?)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

Mapping $\mathcal{ER}^{\mathcal{A}}$ to a relational model

Attributes on Relationships

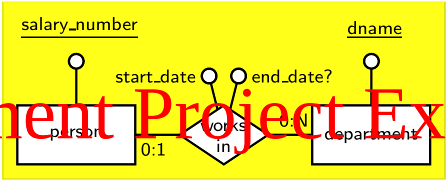
Attributes of a relationship go on the same table as that which implements the relationship

Tables generated from attributes of relationships



person(salary_number,dname,start_date,end_date?)
department(dname)
person(dname) $\overset{fk}{\Rightarrow}$ department(dname)

Quiz 8: Handling of $\mathcal{ER}^{\mathcal{A}}$ 0:1 cardinality



Which is the most precise mapping of the ER schema?

A

person(salary_number)
department(dname)
works_in(salary_number,dname,start_date,end_date?)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

B

person(salary_number)
department(dname)
works_in(salary_number,dname,start_date,end_date?)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

C

person(salary_number,dname,start_date,end_date?)
department(dname)
person(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

D

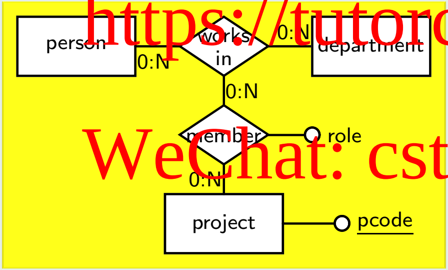
person(salary_number,dname)
department(dname,salary_number,start_date,end_date?)
department(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)

Mapping $\mathcal{ER}^{\mathcal{N}}$ to a relational model

Nested Relationships

If relationship R connects to relationship S, (1) map S as normal, (2) when mapping R, treat S as if it were an entity, and apply the normal rules for mapping R.

Mapping Nested Relationships



person(salary_number)
department(dname)
project(pcode)
works_in(salary_number,dname)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)

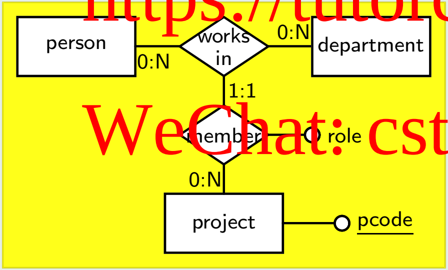
member(pcode,salary_number,dname,role)
member(salary_number,dname) $\stackrel{fk}{\Rightarrow}$ works_in(salary_number,dname)
member(pcode) $\stackrel{fk}{\Rightarrow}$ project(pcode)

Mapping $\mathcal{ER}^{\mathcal{N}}$ to a relational model

Nested Relationships

If relationship R connects to relationship S (1) map S as normal, (2) when mapping R, treat S as if it were an entity, and apply the normal rules for mapping R.

Mapping Nested Relationships



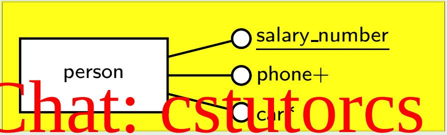
person(salary_number)
department(dname)
project(pcode)
works_in(salary_number,dname,pcode,role)
works_in(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
works_in(dname) $\stackrel{fk}{\Rightarrow}$ department(dname)
works_in(pcode) $\stackrel{fk}{\Rightarrow}$ project(pcode)

Mapping $\mathcal{ER}^{\mathcal{V}}$ to a relational model

Multi-valued Attributes

Each multi-valued attribute $E.A_v$ is stored in its own table RA_v, together with the key attributes of the table R used to represent the entity R.
All attributes of RA_v form the key of RA_v, and there is a foreign key from RA_v to R
No efficient method of representing + constraint

Tables for multi-valued attributes



person(salary_number)
person_phone(salary_number,phone)
person_phone(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)
person_car(salary_number,car)
person_car(salary_number) $\stackrel{fk}{\Rightarrow}$ person(salary_number)

Worksheet: Mapping $\mathcal{ER}^{\mathcal{ADHKLMOSWN}}$ to a relational model

Take your $\mathcal{ER}^{\mathcal{ADHKLMOSWN}}$ *schema in the worksheet, and map it into a relational schema.*