# ER to Relational Mapping

P.J. McBrien

Imperial College London

# Designing a Relational Database Schema

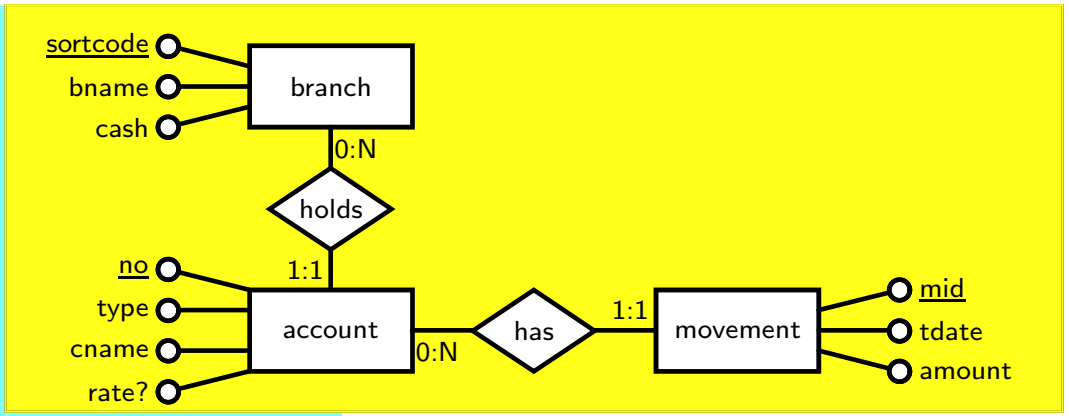
How do you design a relational database schema for a particular UoD?

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
)
```



# Mapping an ER Schema to a Relational Schema

Borrowing the terminology of **object relational mapping** (**ORM**)

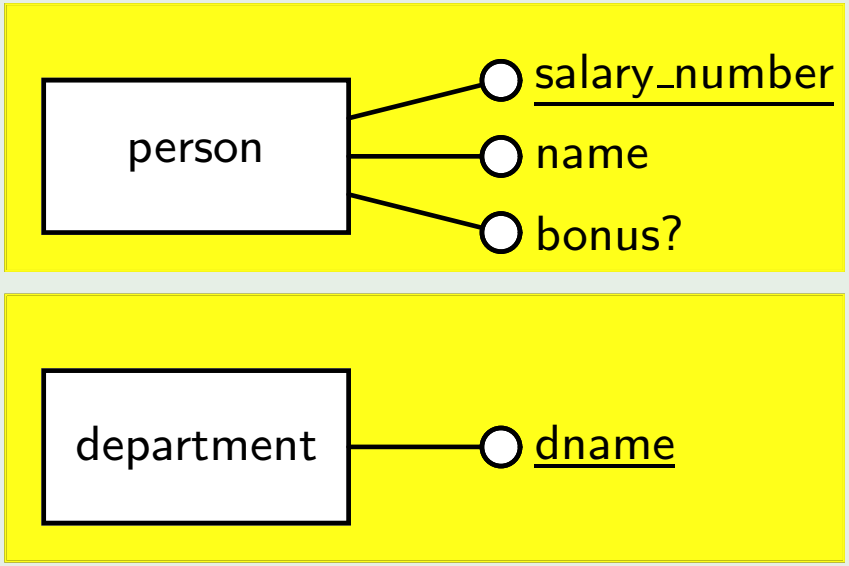
- **table per type** (**TPT**)
  Each ER entity is mapped to a corresponding single relation, with all directly connected attributes and relationships of that entity.
- **table per hierarchy** (**TPH**)
  Each ER entity that is not a subclass of another ER entity is mapped to a single relation with all the data of the subclasses in the same table.
- **table per concrete type** (**TPC**)
  Each ER entity is mapped to a corresponding single relation, with all attributes and relationships of any superclasses included in that relation.
- **table per modelling construct** (**TPM**)
  Store all objects in the ER Schema in a fixed set of tables in a relational model for the various ER modelling constructs.

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

Taking a 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



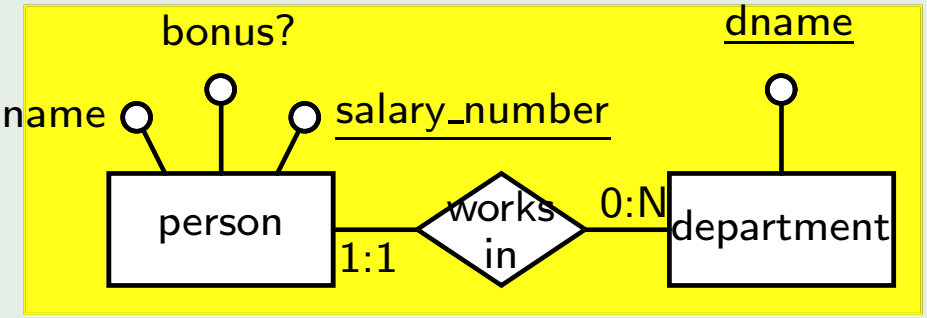

person(<u>salary_number</u>, name, bonus?)
department(<u>dname</u>)

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

Taking a 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 $R\_R_1\_R_2(\vec{K_1}) \rightarrow R_1(\vec{K_1})$
   3. a foreign key $R\_E_1\_E_2(\vec{K_2}) \rightarrow E_2(\vec{K_2})$
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 $R_1(\vec{K_2}) \rightarrow R_2(\vec{K_2})$
   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



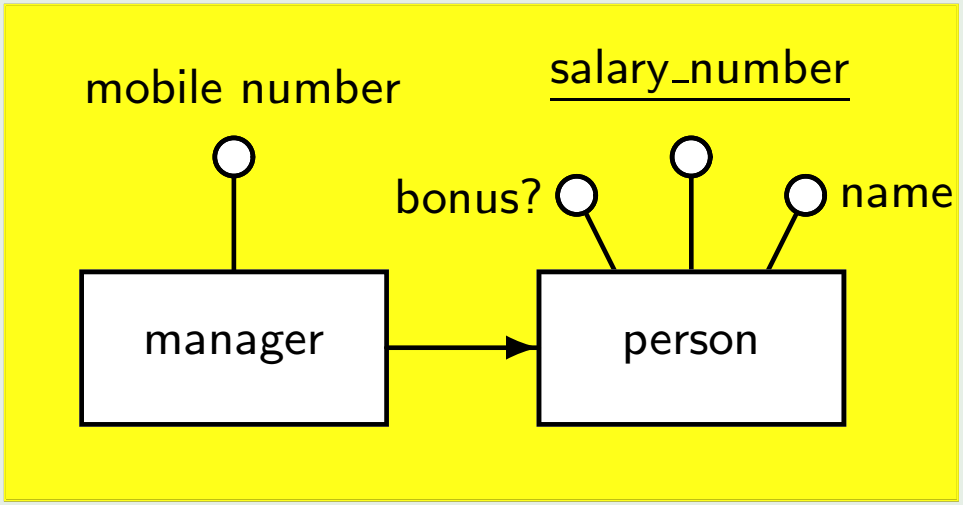

person(<u>salary_number</u>, name, bonus?, dname)
department(<u>dname</u>)
person(dname) $\rightarrow$ department(dname)

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

Taking a 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 $R_s(\vec{K}) \rightarrow R(\vec{K})$

## Tables generated from subsets




person(<u>salary_number</u>, name, bonus?)
manager(<u>salary_number</u>, mobile_phone)
manager(salary_number) $\rightarrow$ person(salary_number)

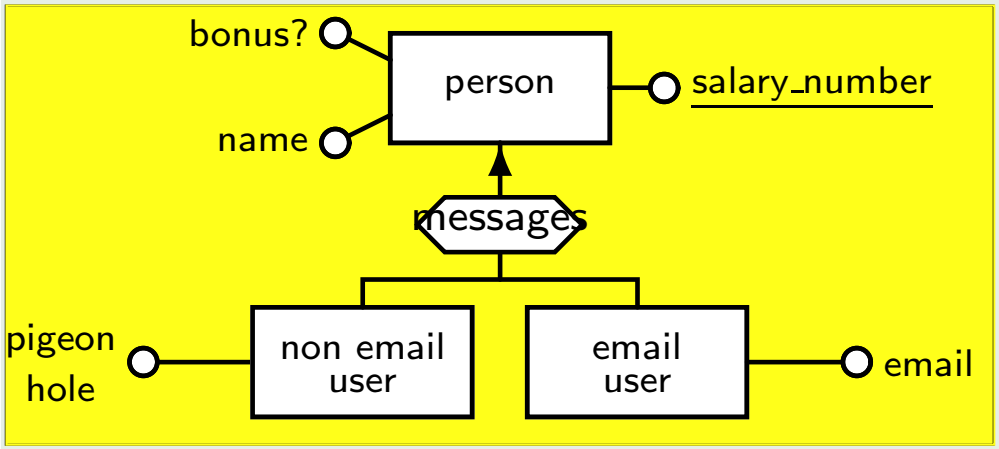
# Mapping $\mathrm{ER}^{\mathcal{KLOS}}$ to a relational model

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

Taking a 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

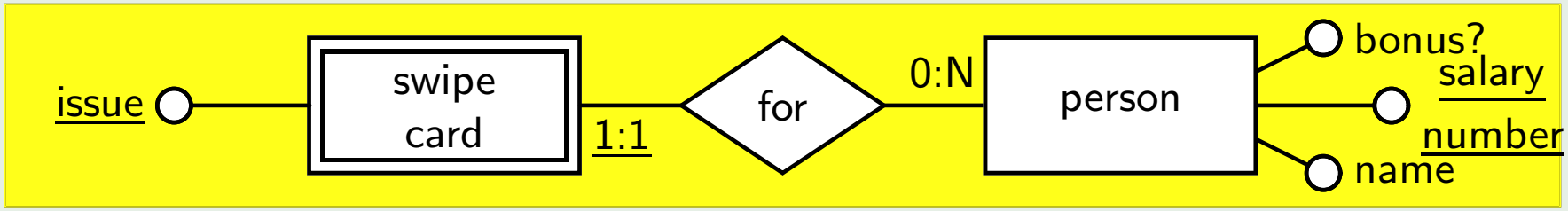



person(<u>salary_number</u>, name, bonus?)
non_email_user(<u>salary_number</u>, pigeon_hole)
non_email_user(salary_number) $\rightarrow$ person(salary_number)
email_user(<u>salary_number</u>, email)
email_user(salary_number) $\rightarrow$ person(salary_number)

# Mapping 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

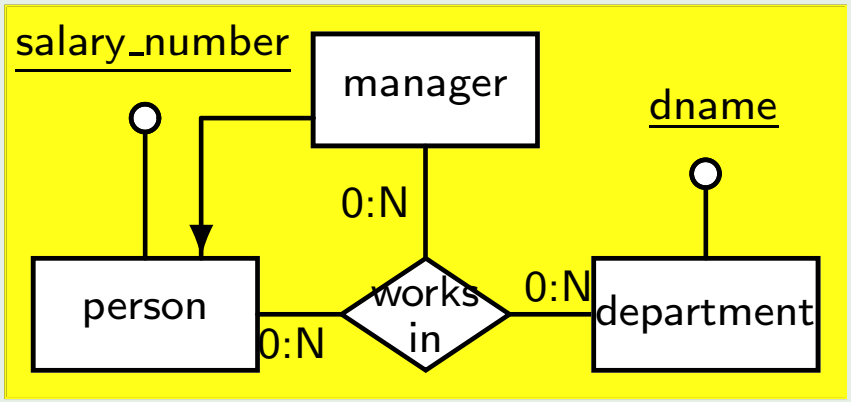



person(<u>salary_number</u>, name, bonus?)
swipe_card(<u>salary_number, issue</u>)
swipe_card(salary_number) $\rightarrow$ person(salary_number)

# Mapping 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

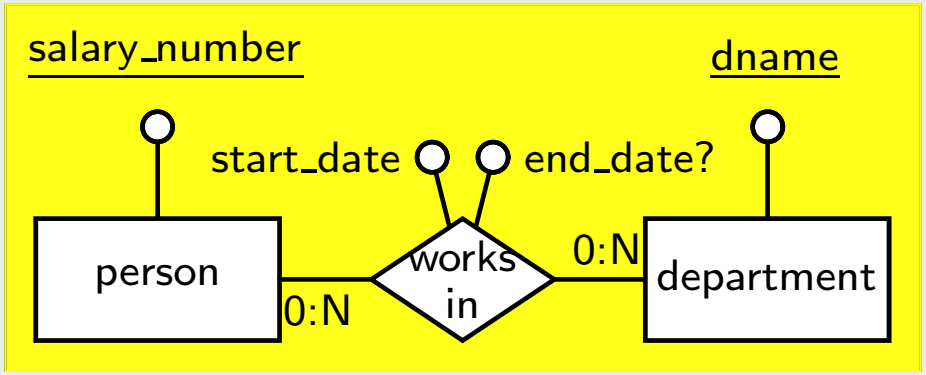



person(salary_number)
manager(salary_number)
manager(salary_number) $\rightarrow$ person(salary_number)
department(dname)
works_in(person_salary_number, manager_salary_number, dname)
works_in(person_salary_number) $\rightarrow$ person(salary_number)
works_in(manager_salary_number) $\rightarrow$ manager(salary_number)
works_in(dname) $\rightarrow$ department(dname)

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

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

## Tables generated from attrbutes of relationships




person(<u>salary_number</u>)
department(<u>dname</u>)
works_in(<u>salary_number, dname</u>, start_date, end_date?)
works_in(salary_number) $\longrightarrow$ person(<u>salary_number</u>)
works_in(dname) $\longrightarrow$ department(<u>dname</u>)

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