Database Systems 2

Lecture 3

Enhanced ER Modelling
Problems with ER Models

Enhanced Entity Relationship Models

Most situations can be modelled using the techniques already described.

However, since the 1980s, there has been a rapid increase in the use of databases in the fields of CAD, CAM and multimedia applications, as well as the rise of online data storage.

This has given rise to the need of additional data modelling techniques which are given the collective name of Enhanced Entity Relationship (EER) Modelling

EER Models cont...

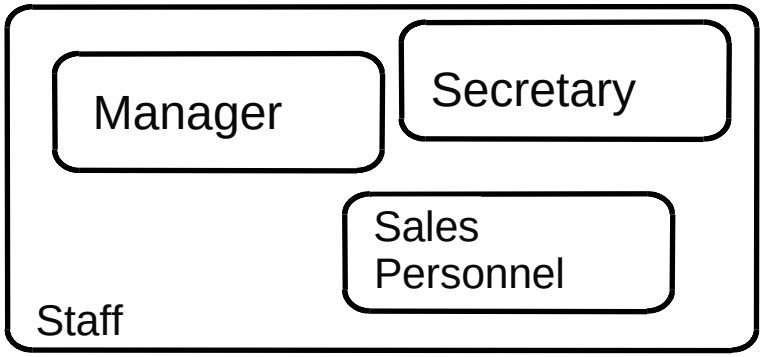
First we need some new entity constructs.

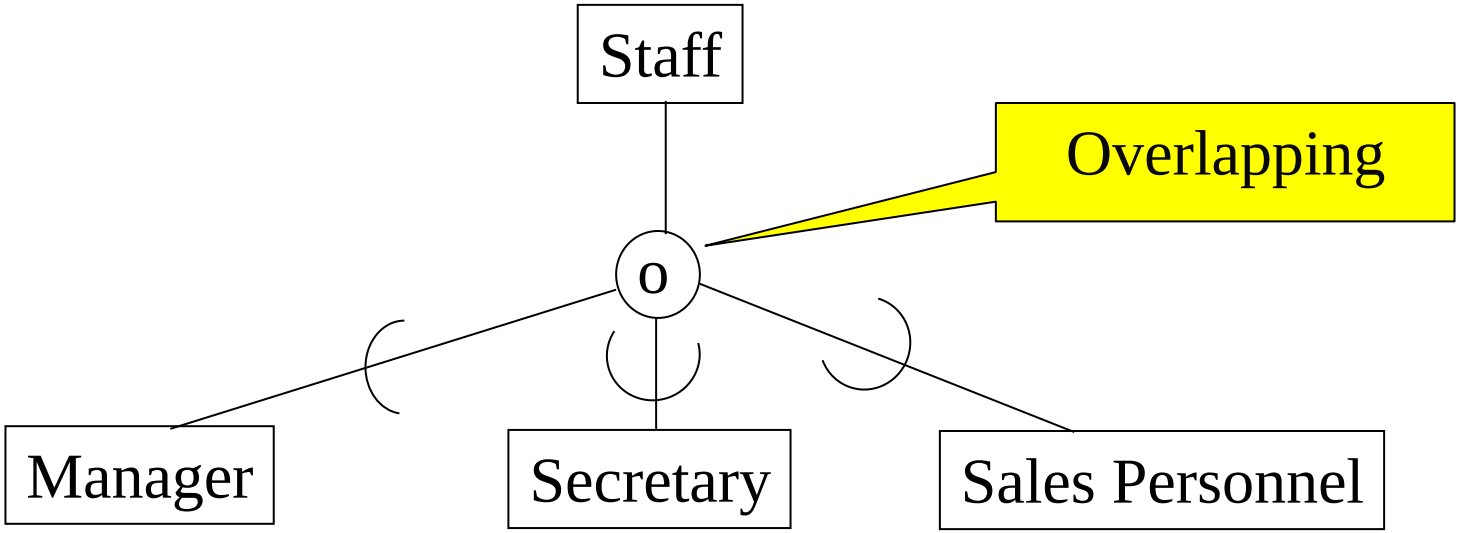
Superclass

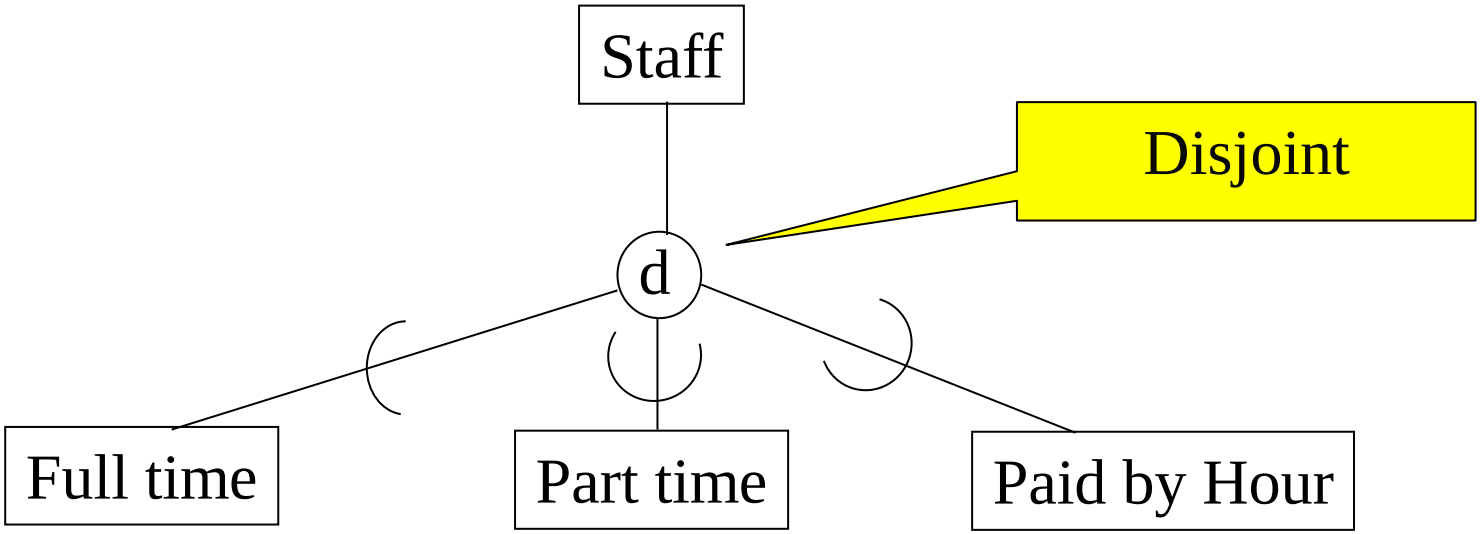
- an entity type that includes distinct subclasses that require to be represented in a data model.

Subclass

- an entity type that has a distinct role and is also a member of a superclass.




Staff
Overlapping
o
Manager
Secretary
Sales Personnel


Staff
Disjoint
d
Full time
Part time
Paid by Hour

EER Models cont…

Subclasses need not be mutually exclusive; a member of staff may be a manager and a sales person.

The purpose of introducing superclasses and subclasses is to avoid describing types of staff with possibly different attributes within a single entity.

This could waste space and you might want to make some attributes mandatory for some types of staff but other staff would not need these attributes at all.

If the subclasses are overlapping

All staff have StaffNo, name, address, dob, bonus

Secretaries also have WPSkills
Managers also have Bonus
All Sales Personnel have SalesArea and Allowance

It is possible for someone to be a manager and sales personnel, or a Secretary and Sales Personnel, or just Sales personnel.

StaffNo	Name	Address	DoB	Bonus	WPSkills	SalesArea	Allowance
S001	Andrews	1 Road	1/1/12	500	Null	London	500
S002	Brown	2 Street	2/2/22	Null	300	Leeds	200
S003	Clark	3 Lane	3/3/33	Null	200	London	300
S004	Davies	4 Avenue	4/4/44	Null	Null	Derby	100

Specialisation

This is the process of maximising the differences between members of an entity by identifying their distinguishing characteristics.

This is the identification of specialised subclasses from the original superclass. This is the process of identifying the different attributes and relationships.

Top-down analysis.

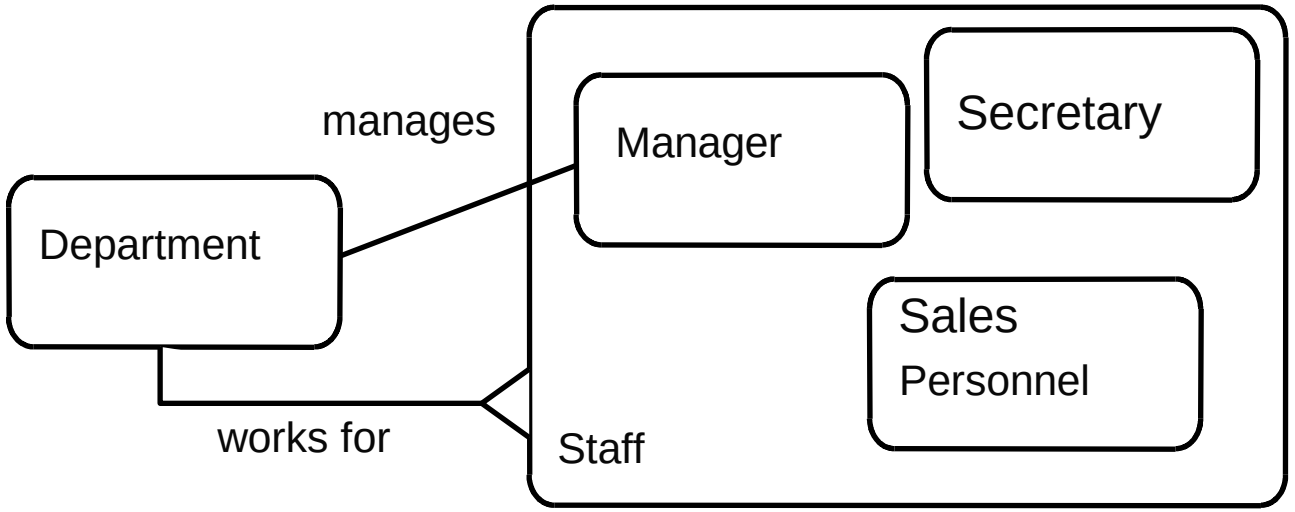
Generalisation

Generalisation is the process of minimising the differences between entities by identifying common features.

This is the identification of a generalised superclass from the original subclasses. This is the process of identifying the common attributes and relationships.

Bottom-up analysis.

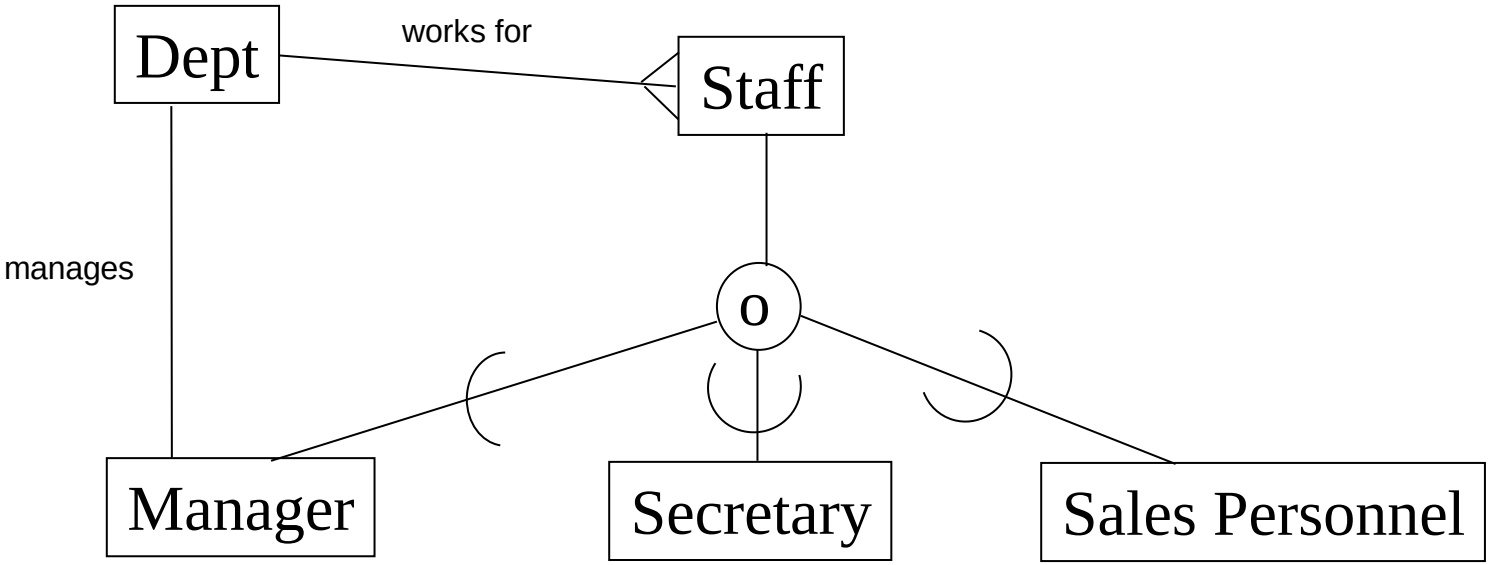
Specialisation example



Starting with a Staff entity, we realise that the manages relationship is only applicable to the Manager subclass, whereas the works_for relationship is applicable to all staff.

We therefore treat the Staff entity as a superclass, and break it down into subclasses.

Showing super/sub classes on an ERD.



Note that subclasses can themselves have subclasses.

The relationship between a superclass and a subclass is always 1 to 1, and so the cardinality is not usually shown.

Implementing Superclasses & Subclasses

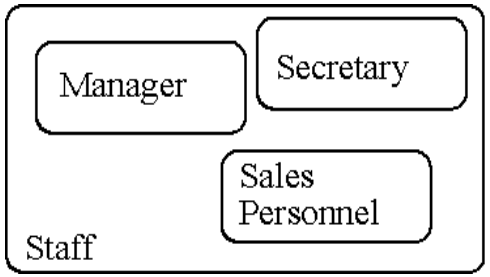
There are three ways of implementing superclasses and subclasses and it depends on the application which will be the most suitable.

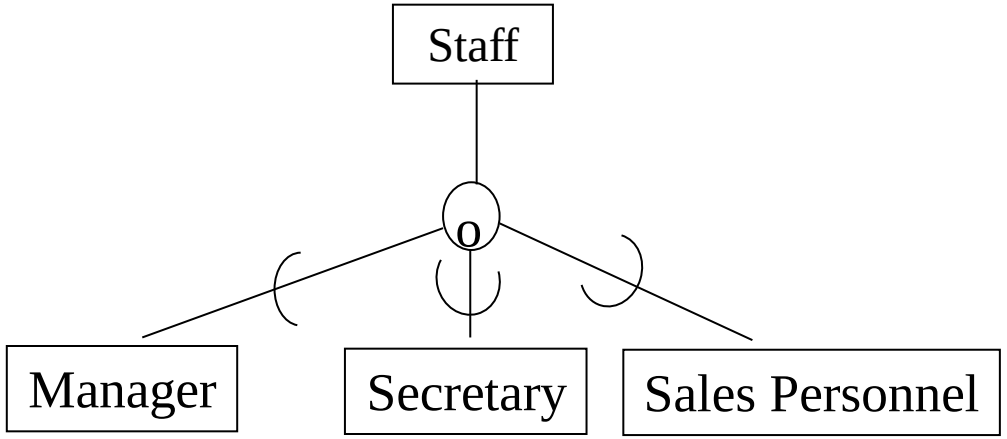
1. One relation for each subclass.

2. One relation for the superclass.

3. One relation for the superclass and one relation for each subclass.

Example





Staff fall into three categories.

They are overlapping.

eg Member of staff could be a manager and also belong to sales personnel.

One relation for each subclass

One relation for each subclass:

```
Manager(staff_no, name, address, dob, bonus)
Secretary(staff_no, name, address, dob, wp_skills)
Sales_personnel(staff_no, name, address, dob, sales_area, allowance)
```

- All attributes are mapped into each subclass. Equivalent to having three separate entity types and no superclass.
- It is useful if no overlap and no relationships between superclass and other entity types.
- Poor if subclasses are not disjoint – causes data duplication leading to problems with consistency.

One relation for the superclass

One relation for the superclass:

```
Staff(staff_no,name,address,dob, bonus,
                wp_skills, sales_area, allowance)
```

- This represents a single entity type with no subclasses.
- This is no good if the subclasses are not disjoint or if there are relationships between the subclasses and the other entities.
- In addition, there will be many null fields if the subclasses do not overlap a lot. However, it avoids any joins to get additional information about each member of staff.

One Relation for each

One relation for the superclass and one relation for each subclass:

Staff(staff_no, name,address,dob)

Manager(*staff_no, bonus)**
Secretary(*staff_no, wp_skills)**
Sales_personnel(*staff_no, sales_area, car_allowance)**

The primary key of the superclass is mapped into each subclass and becomes the subclasses primary key – as well as acting as a foreign key.

This represents most closely the EER model. However is can cause efficiency problems as there needs to be a lot of joins if the additional information is often needed for all staff.

Problems with ER Models

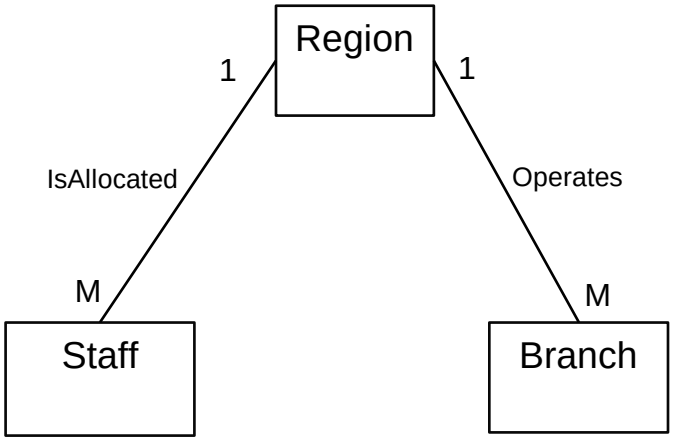
These are called 'connection traps'.

- Fan traps
- Chasm traps

Sometimes, a connection trap may not be significant to the enterprise, and so can be ignored.

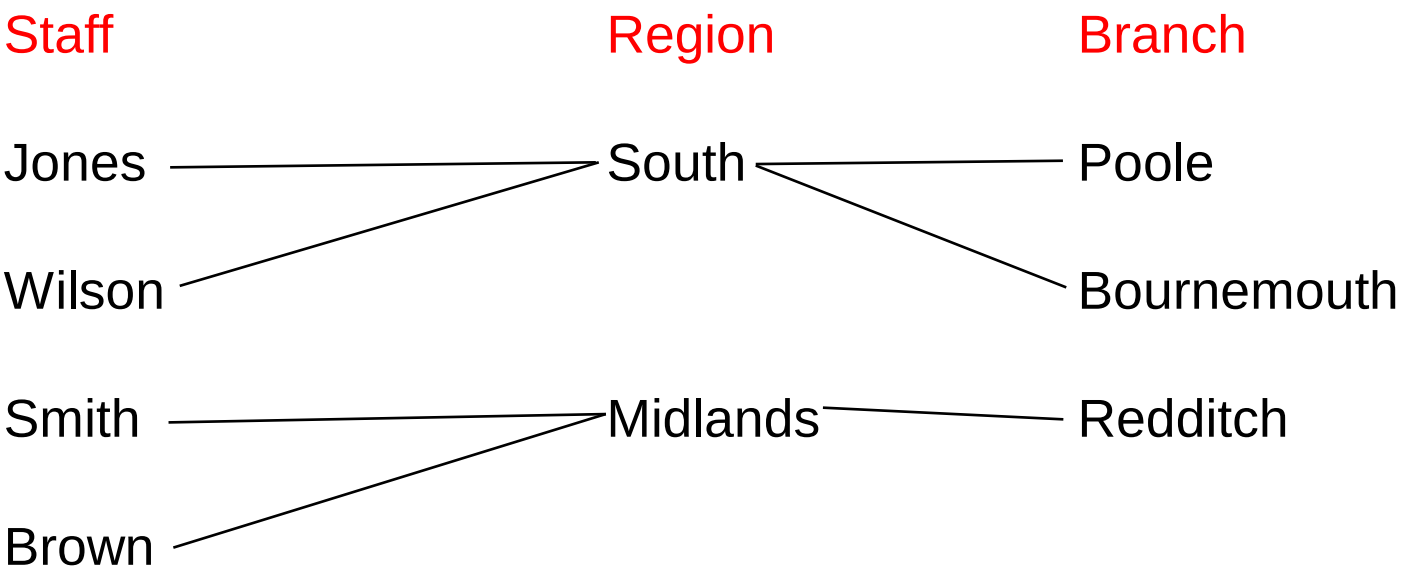
At other times, they may necessitate the restructuring of the conceptual model.

Fan Traps

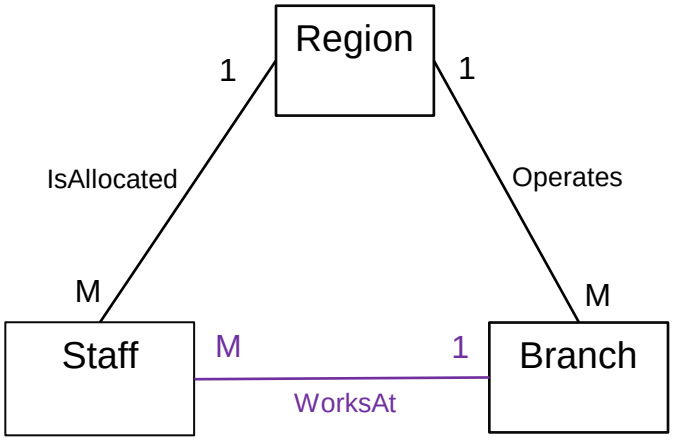


A company has several regional offices.
Each region contains many branch offices.
Each region also employs many staff.

What question can we not answer?

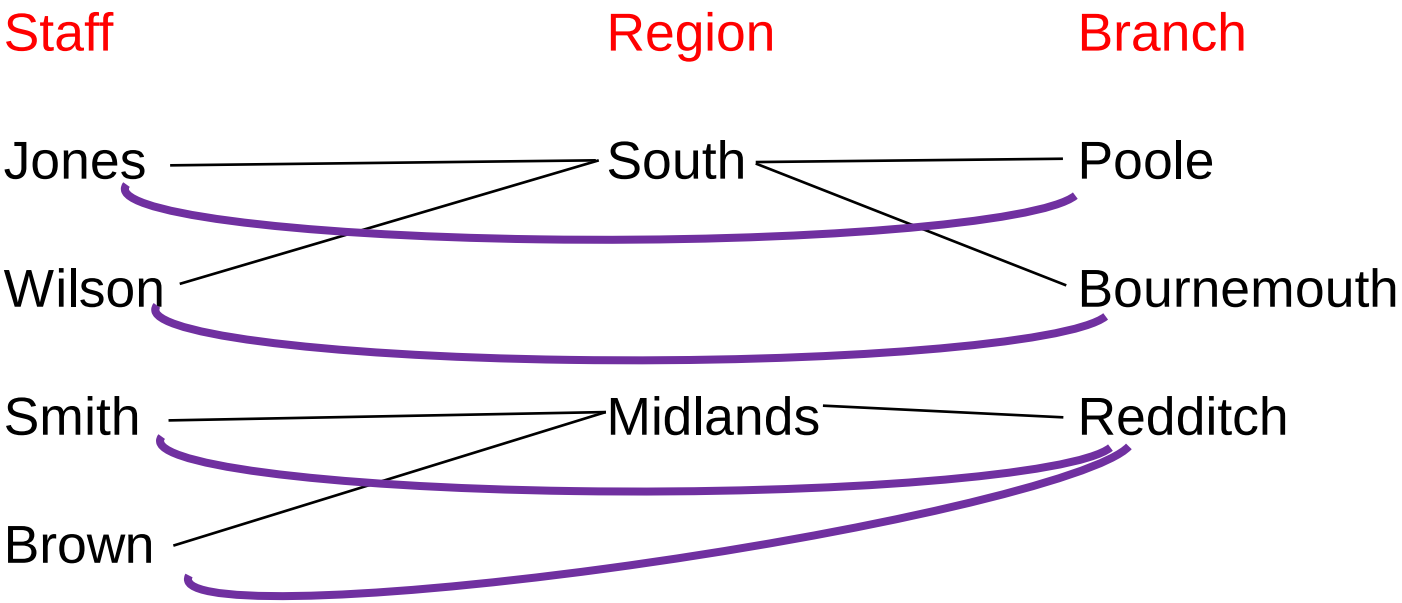


One possible solution

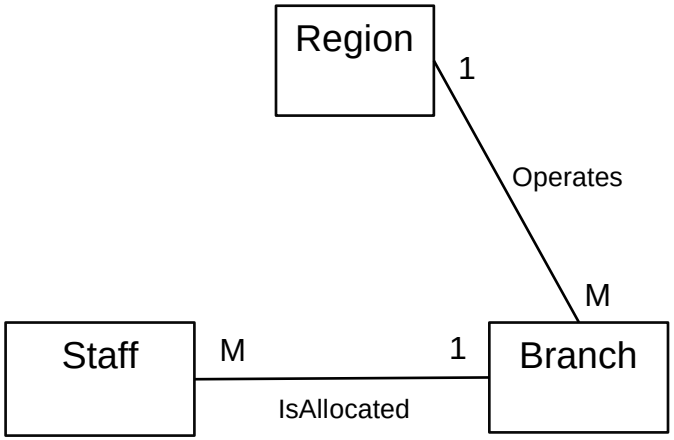


Supply the missing relationship which links Branch to Staff.

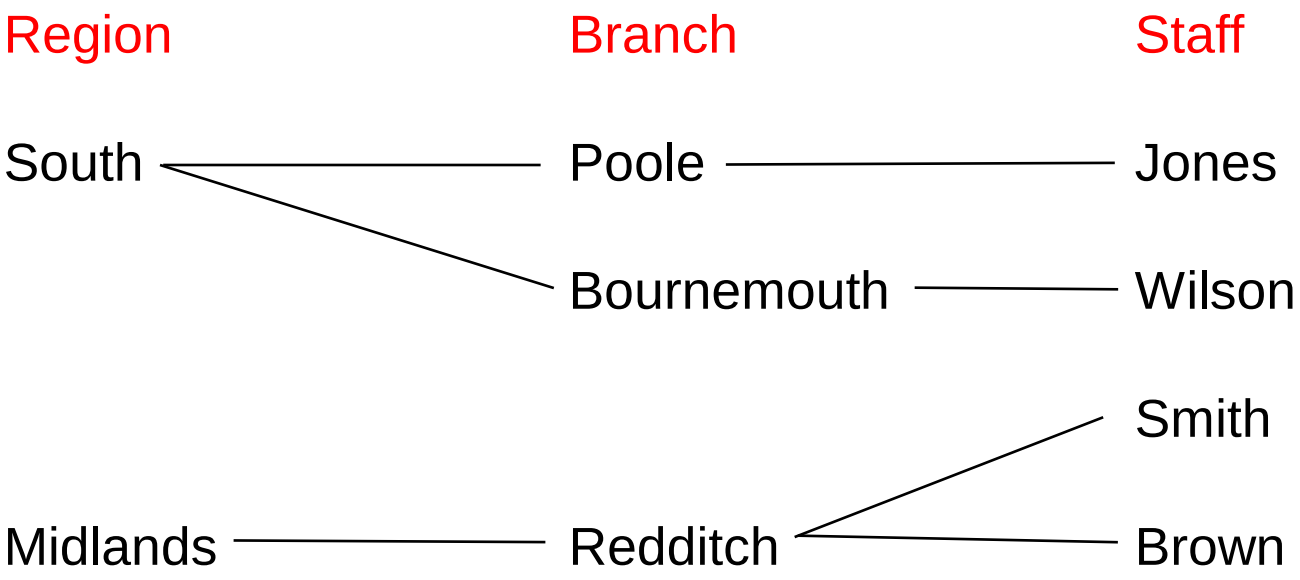
Is there a better way?



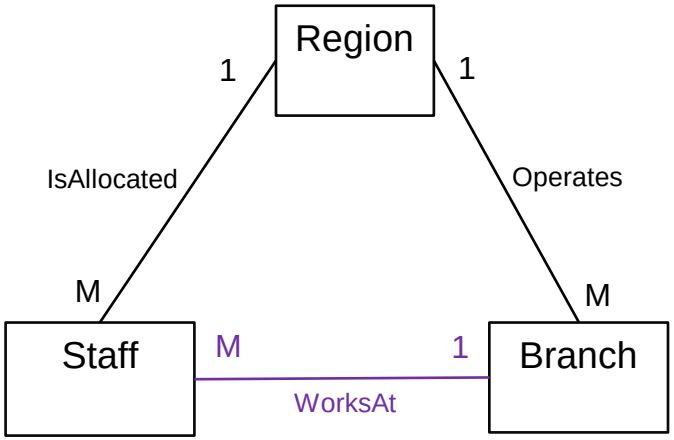
A better solution



Restructure the ERD so that it better reflects the structure of the company.

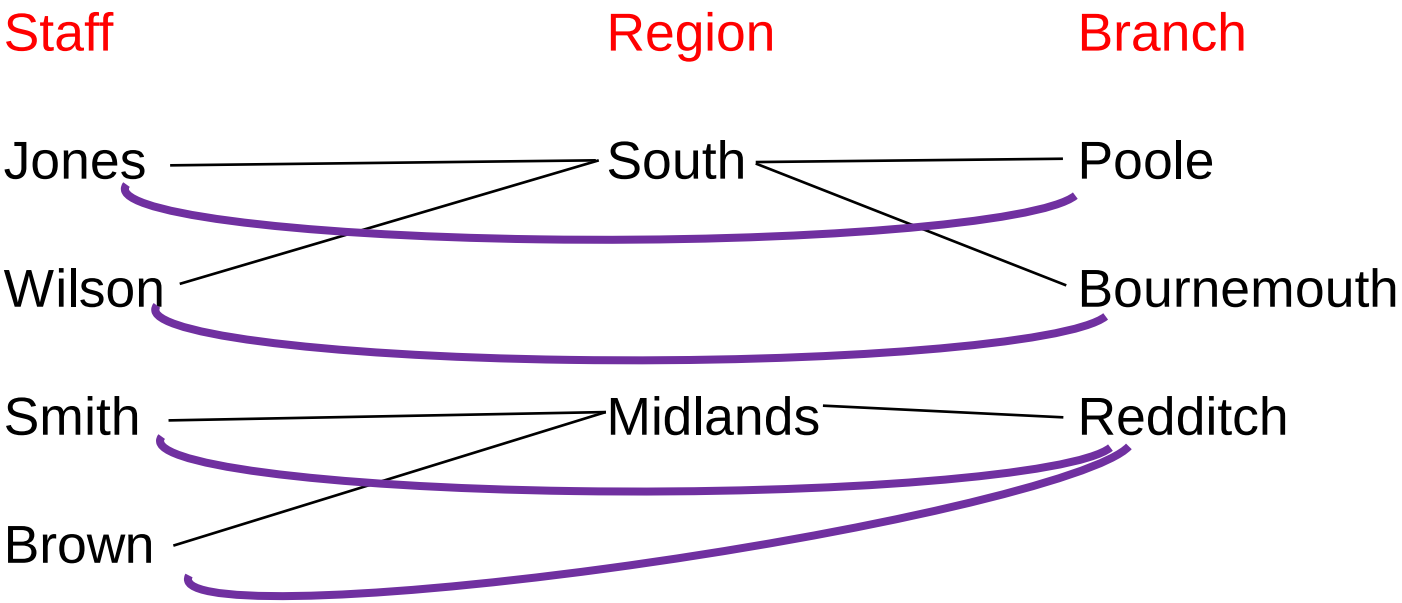


Question

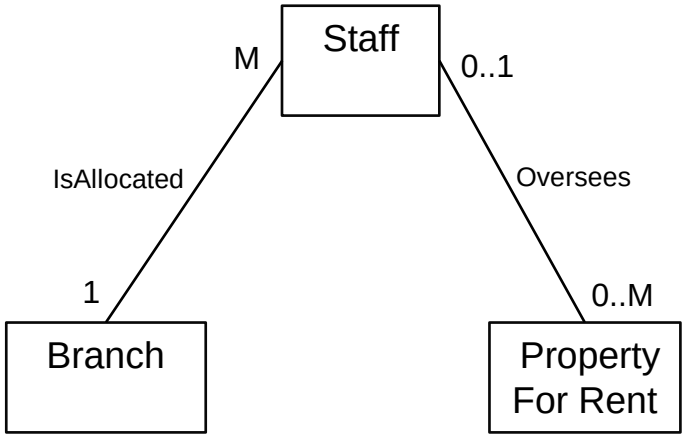


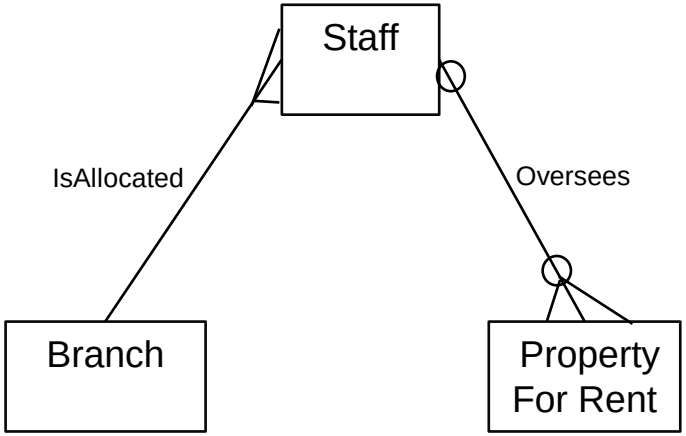
Could we have got rid of the other relationship?

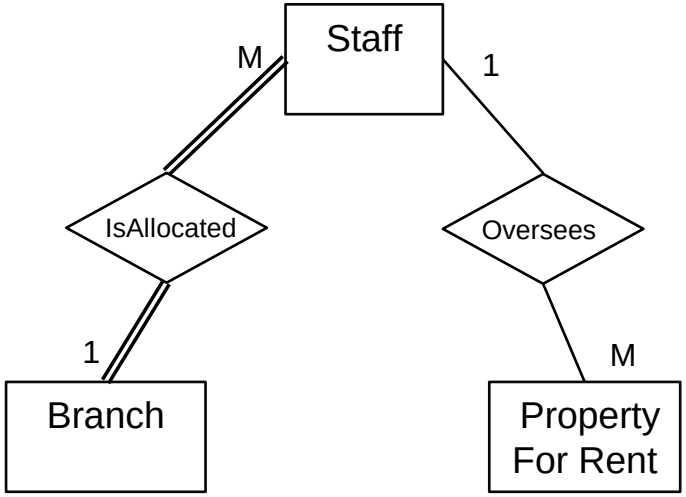
Have we lost any information?



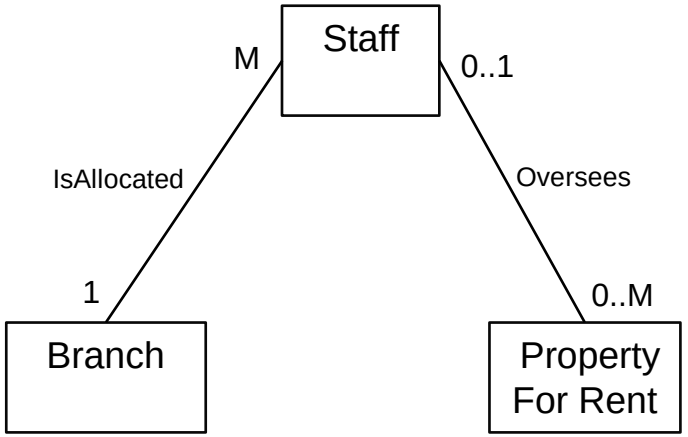
Chasm Traps


Staff
M
0..1
IsAllocated
Oversees
1
0..M
Branch
Property
For Rent


Staff
IsAllocated
Oversees
Branch
Property
For Rent


Staff
M
1
IsAllocated
Oversees
1
M
Branch
Property
For Rent

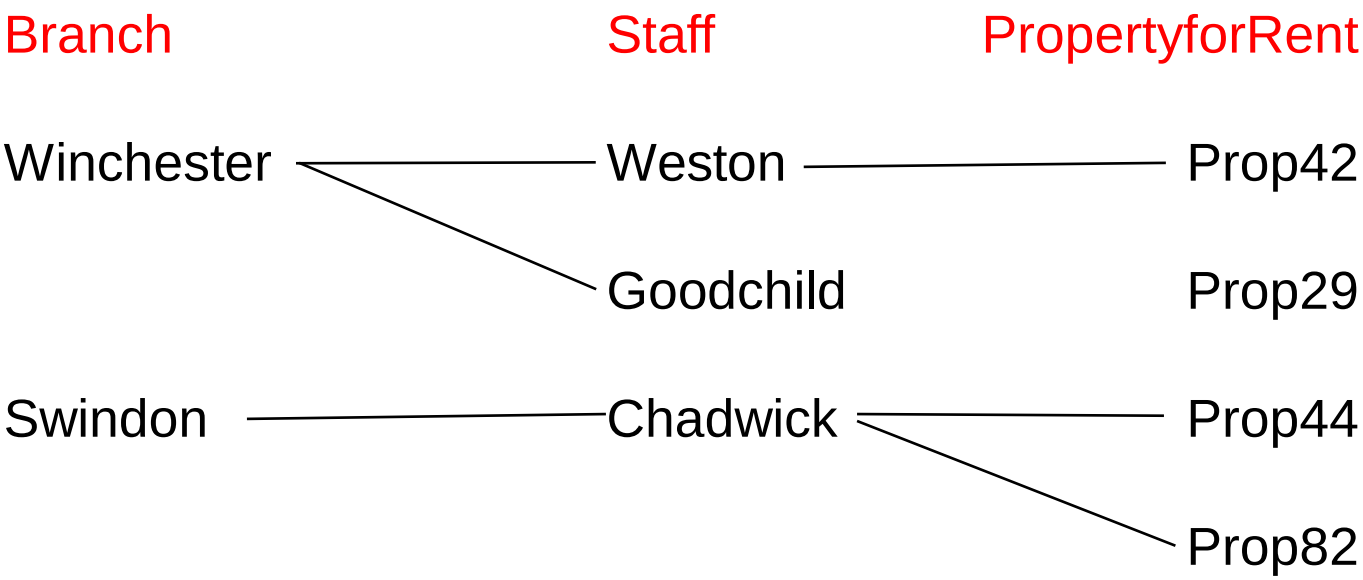
Chasm Traps



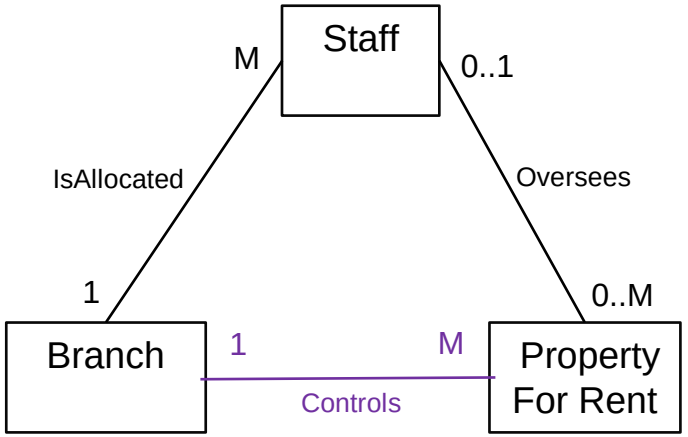
Which Staff member is not participating in the 'oversees' relationship?

Which Property is not participating in the 'oversees' relationship?

What question can we not answer?



Solution



Put in a relationship which links Branch and Property directly.