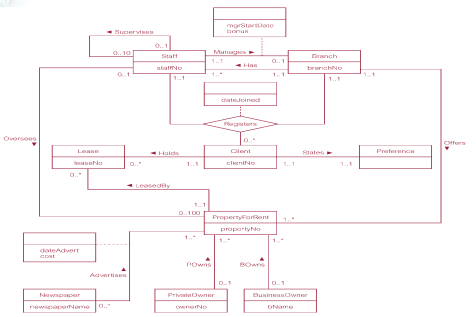
# Chapter 5. Entity Relationship Modeling

**Introduction**
**Concepts of the ER Model**
**Structural Constraints**
**Problems with ER Models**



## ER diagram of Branch user views of *DreamHome*






## Introduction

- high-level conceptual data model to facilitate database design.
- developed by Peter Chen (1976).
- purpose :-
  a) support a user's perception of data.
  b) to conceal the more technical aspects associated with database design.
  c) identify the processes and constraints.
  d) to implement the database.
  e) produce consistent and non-redundancy model.



- used to describe data in external level.
- independent of the particular DBMS and hardware platform.
- using UML (Unified Modeling Language) notations.
- **UML** → modeling language created to represent project areas developed using the object-oriented method.
- extended ER Model → Enhanced Entity Relationship Modeling (EER).



## Concepts of the ER Model

- based on a real event in an organization.
- 3 basic concepts of E-R model :-
  a) entity types
  b) relationship types
  c) attributes



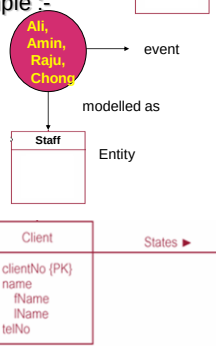
## Entity types

***"group of objects with same properties, identified by enterprise as having an independent existence."***

- represents a set of 'objects' in the 'real world' (physical) or conceptual (abstract) with same properties.
- **ENTITY** – an object or concept that is uniquely identifiable.
- important objects in information system.

| Physical existence | |
|---|---|
| Staff | Part |
| Property | Supplier |
| Customer | Product |
| **Conceptual existence** | |
| Viewing | Sale |
| Inspection | Work experience |

- **entity notation →**
- example :-

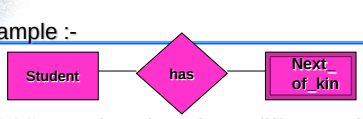




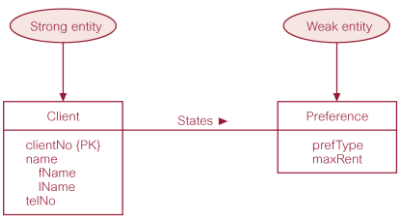

- divided to 2 types of entities :-
  - a) ***strong entity type***
    - an entity type that is *not* existence-dependent on some other entity type.
    - example : employee, student, patient, client
    - known as parents or dominant.
  - b) ***weak entity type***
    - an entity type that is existence-dependent on some other entity type.
    - known as child or subordinate.
    - example : next_of_kin, preferences



- example :-




- in UML notation, there is no difference between strong and weak entities.





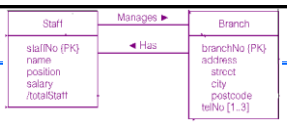

## Relationship Types

***"set of meaningful associations among entity types."***

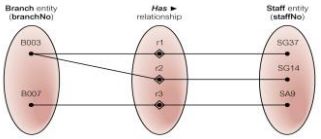
- ***relationship* occurrence – *uniquely identifiable association, which includes one occurrence from each participating entity type.***
- each relationship is given a name that describe its function and shown as a line connecting the associated entity types.
- a relationship is named using a verb (supervises or manages) or short phrase including a verb (leasedBy) written on the top of the relationship line.



- example :

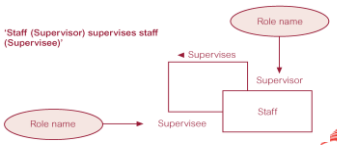



- relationship above known as '*HAS-A' relationship*.
- shown using the semantic net.
- semantic net – an object-level model, which uses the symbol • to represent entities and the symbol ◆ to represent relationships.

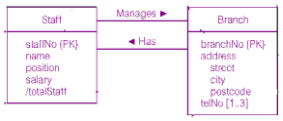




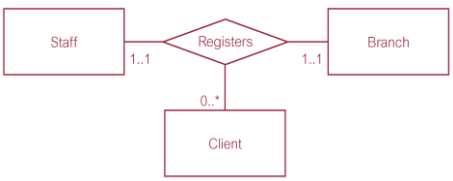

- Degree of a Relationship
  ***Number of participating entities in relationship.***
- Relationship of degree :
  - ***recursive relationship***
    ***relationship type where same entity type participates more than once in different roles.***



- **two is binary**

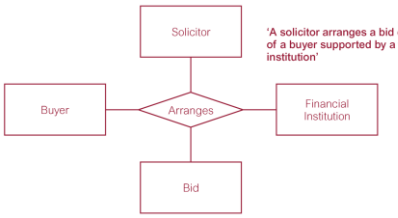



- **three is ternary**



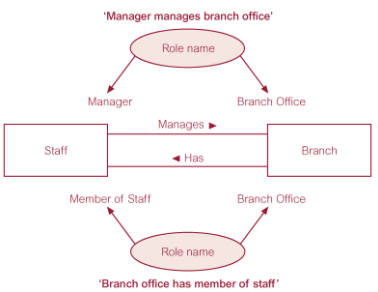



- **four is quaternary.**






## Entities associated through two distinct relationships with role names

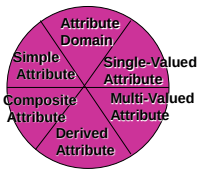





## Attributes

"*a property of an entity or a relationship type.*"

- the name given must be unique.
- example : name, icNo, address, dateOfbirth etc.
- 5 types of attributes:-






## Attribute Domain

"**set of allowable values for one or more attributes. "**

- example: domain for staffNo is integer range from 1 - 5000; domain for name is text 1 - 25.
- different attribute can share same domain.
- example: name and department
- domain for dateOfbirth
  → subdomain : day, months, year.



## Simple Attribute

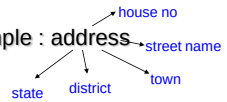
"*attribute composed of a single component with an independent existence.*"

- example : sex, name, salary etc.
- known as atomic attributes.

## Composite Attribute

"*attribute composed of multiple components, each with an independent existence.*"

- example : address






## Derived Attribute

"*attribute that represents a value that is derivable from value of a related attribute or set of attributes, not necessarily in the same entity type.*"

- symbol used is ' / ' before the attribute name.
- example : age can be derived from date of birth

Derived attribute — Basic attribute



## Multi-valued Attribute

"*attribute that holds multiple values for each occurrence of an entity type.*"

- have a set of numbers with upper and lower limit.
- put in a square bracket at the end of the attribute's name.
- example :- no_tel [1..3]



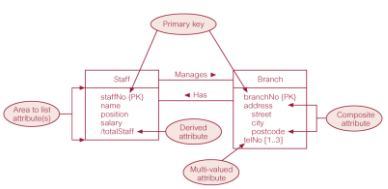
## Single-valued Attribute

"*attribute that holds a single value for each occurrence of an entity type.*"

- can consists of simple attribute/composite.
- example : studentNo, icNno, staffNo etc.



- using the UML notation, the attributes' name are written in a rectangle below the entity's name.
- for attribute chosen as a primary key, will be tagged as PK at the end of the attribute's name.
- also known as key attribute.
- other tags such as : FK (*Foreign Key*), AK(*Alternate Key*)





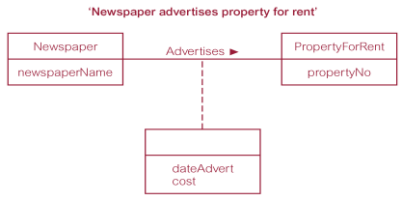

## Tutorial

| Student |
|---|
| studNo {PK} |
| name |
| icNo {AK} |
| dob |
| address |
| houseNo |
| street |
| city |
| telNo [1..3] |
| /age |
| progID {FK} |

**Primary Key -**
**Alternate Key -**
**Foreign Key -**
**Composite Attribute -**
**Derived Attribute -**
**Multi-Valued Attribute -**

## Attributes on Relationships

- can have attribute to describe the relationship.
- represented by dotted line.






## Relational keys

"***a data item that allows us to uniquely identify individual occurrences of an entity type***."

- the value must be unique and not null.
- divided to 3 type of keys :-

a) ***candidate key***

- ***minimal set of attributes that uniquely identifies each occurrence of an entity type.***
- example: branchNo, staffNo and icNo.

b) ***primary key***

- ***an entity type may have one or more possible candidate keys, one of which is selected to be the primary key.***
- example: staffNo {PK} and icNo {AK}



c) ***composite key***

- **attribute composed of multiple components, each with an independent existence.**
- example :

Advertisement (property_no, newspaper_name, date, cost)

property_no, newspaper_name, date

composite keys



## Structural Constraints

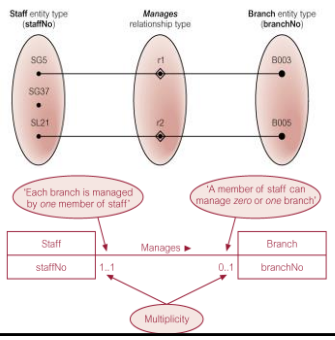
- main type of constraint on relationships is called multiplicity.
- multiplicity - ***number (or range) of possible occurrences of an entity type that may relate to a single occurrence of an associated entity type through a particular relationship***.
- stated in a form of numbers or range at the end of the relationship.
- represents policies (called business rules) established by user or company.



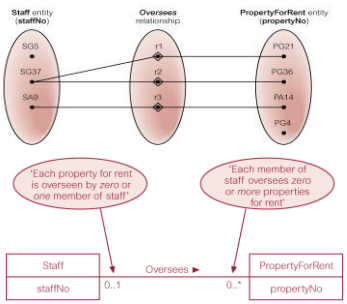
- the most common degree for relationships is binary.
- binary relationships are generally referred to as being:
  - one-to-one (1:1)
  - one-to-many (1:*)
  - many-to-many (*:*)



## Semantic net of Staff *Manages* Branch relationship type

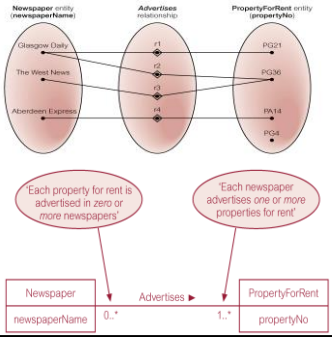


## Semantic net of Staff *Oversees* PropertyForRent relationship type






## Semantic net of Newspaper *Advertises* PropertyForRent relationship type






## Summary of multiplicity constraints

| Alternative ways to represent multiplicity constraints | Meaning |
|---|---|
| 0..1 | Zero or one entity occurrence |
| 1..1 (or just 1) | Exactly one entity occurrence |
| 0..* (or just *) | Zero or many entity occurrences |
| 1..* | One or many entity occurrences |
| 5..10 | Minimum of 5 up to a maximum of 10 entity occurrences |
| 0, 3, 6–8 | Zero or three or six, seven, or eight entity occurrences |



- multiplicity is made up of two types of restrictions on relationships: ***cardinality*** and ***participation***.
- ***cardinality***
  - describes maximum number of possible relationship occurrences for an entity participating in a given relationship type.
- most common degree for relationship - binary.
- ***cardinality ratio*** – determine the number of possible relationships for each participating entity.
- ratio :- 1:1 , 1:* and *:*



## *participation*

- determines whether all or only some entity occurrences participate in a relationship.
- divided to 2 parts :

***i) total (mandatory)***

*- if an entity's existence requires the occurrence of an associated entity in a particular relationship.*

- will be represented by value of 1.

*- participation a set of entity E in a set of relationship H is total if each entity in E participates in at least one relationship of H.*

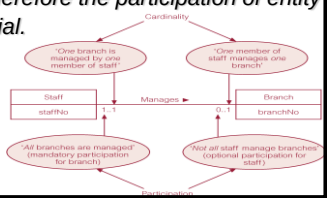


## *ii) partial (optional)*

*- the existence of an entity does not require the occurrence of an associated entity in a particular relationship.*

*- will be represented by value of 0.*

*- if only parts of the entity in E participates in relationship H, therefore the participation of entity set E in H is partial.*



## Problems with ER Models

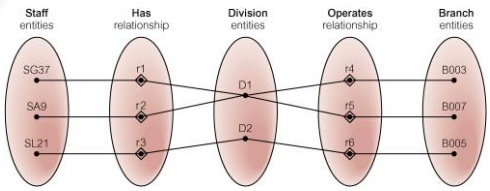
- problems may arise when designing a conceptual data model called ***connection traps***.
- often due to a misinterpretation of the meaning of certain relationships.
- two main types of connection traps are called ***fan traps*** and ***chasm traps***.
- **Fan Trap**
  - where a model represents a relationship between entity types but pathway between certain entity occurrences is ambiguous.
  - exists where two or more 1:* relationships fan out from the same entity.



## An Example of a Fan Trap

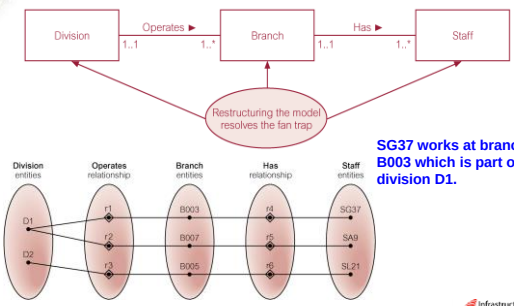

- there are 2 1:* relationships (Has and Operates) emanating from the same entity called Division. problem arises when we want to know which members of staff work at a particular branch.
- examine some occurrences of the Has and Operates relationships using values for the primary key attributes of the Staff, Division and Branch entity types.

## Semantic Net of ER Model with Fan Trap




- At which branch office does staff number SG37 work?

- resolve the problem by ***restructuring the original ER model to represent the correct association between the entities.***



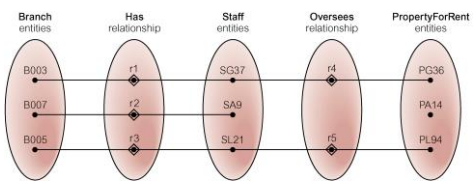

**SG37 works at branch B003 which is part of division D1.**

## Chasm Trap

- where a model suggests the existence of a relationship between entity types, but pathway does not exist between certain entity occurrences.
- occur when there are one or more relationships with a minimum multiplicity of zero forming part of the pathway between related entities.
- example :-

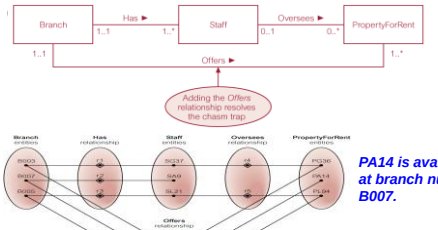

- problem occurs when we want to know which properties are available at each branch.

- examine some occurrences of the Has and Oversees relationships using values for the primary key attribute of the Branch, Staff and PropertyForRent entity types.




- At which branch office is property PA14 available?

to solve the problem, ***need to identify the missing relationships between two entities which in this case is the Offers relationship between the Branch and PropertyForRent.***






***PA14 is available at branch number B007.***