2020-2021

5COSC002W DATABASE SYSTEMS
Lecture 03

LOGICAL DATABASE DESIGN
Mapping a conceptual ER model to a logical ER model

UNIVERSITY OF WESTMINSTER

UNIVERSITY OF TECHNOLOGICAL INNOVATION WESTMINSTER

Lecture 03 – Outline

- Relational model and relational keys
- Use of tables to represent data.
- Mapping relationships from conceptual to logical
  1) *One-to-many*
  2) *One-to-one mandatory on both sides*
  3) *One-to-one optional on one side*
  4) *One-to-one optional on both sides*
  5) *Many-to-many*
  6) *Complex relationships: ternary and quaternary*
  7) *Generalisation with {Mandatory, And}*
  8) *Generalisation with {Optional, And}*
  9) *Generalisation with {Mandatory, Or}*
  10) *Generalisation with {Optional, Or}*

UNIVERSITY OF TECHNOLOGICAL INNOVATION WESTMINSTER

Database Design Methodology – Step 2

LOGICAL DESIGN

Produce a Logical Data Model i.e. relational schema

Construct a model of the data used in a firm

based on specific data organisation (here relational schema)

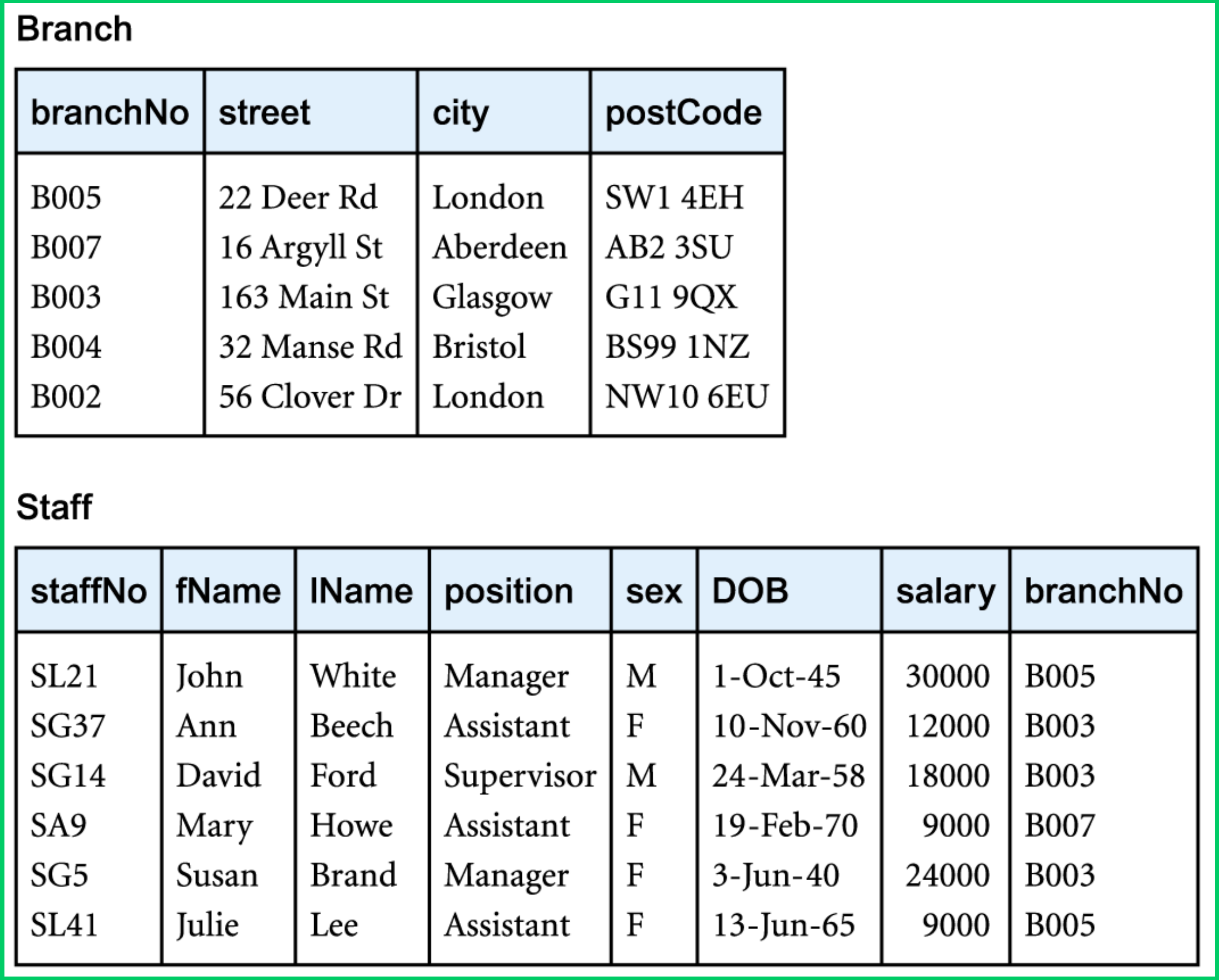
independent of DBMS & other physical considerations

- Step 2.1 Derive relations (i.e. tables) for logical data model
- Step 2.2 Validate relations using normalization
- Step 2.3 Validate relations against user transactions
- Step 2.4 Define integrity constraints
- Step 2.5 Review logical data model with user
- Step 2.6 Merge logical data models into global model
- Step 2.7 Check for future growth

UNIVERSITY OF TECHNOLOGICAL INNOVATION WESTMINSTER

Relational Model

Building Block = relation = table



Branch

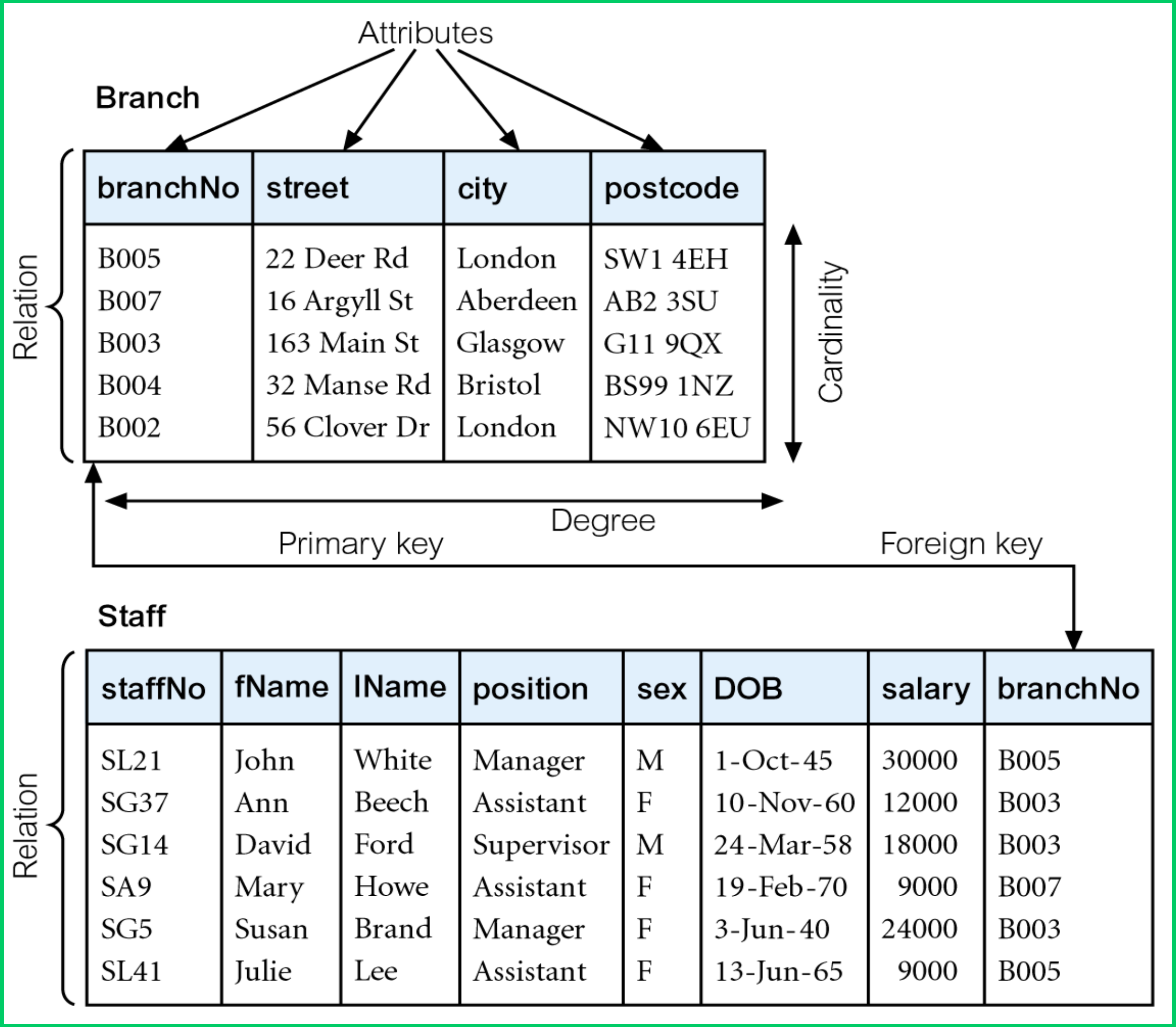
branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Relational Model

Interconnected relations



Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

UNIVERSITY OF TECHNOLOGICAL INNOVATION WESTMINSTER

Keys

- Candidate Key
  - Minimal set of attributes that uniquely identifies each occurrence of an entity.
- Primary Key
  - Candidate key selected to uniquely identify each occurrence of an entity.
- Compound Key
  - A candidate key that consists of two or more attributes.
  - Each attribute that makes up the compound key is a **simple key** in its own right.
- Composite Key
  - A candidate key that consists of two or more attributes.
  - At least one attribute that makes up the composite key is not a **simple key** in its own right.

Relational Keys

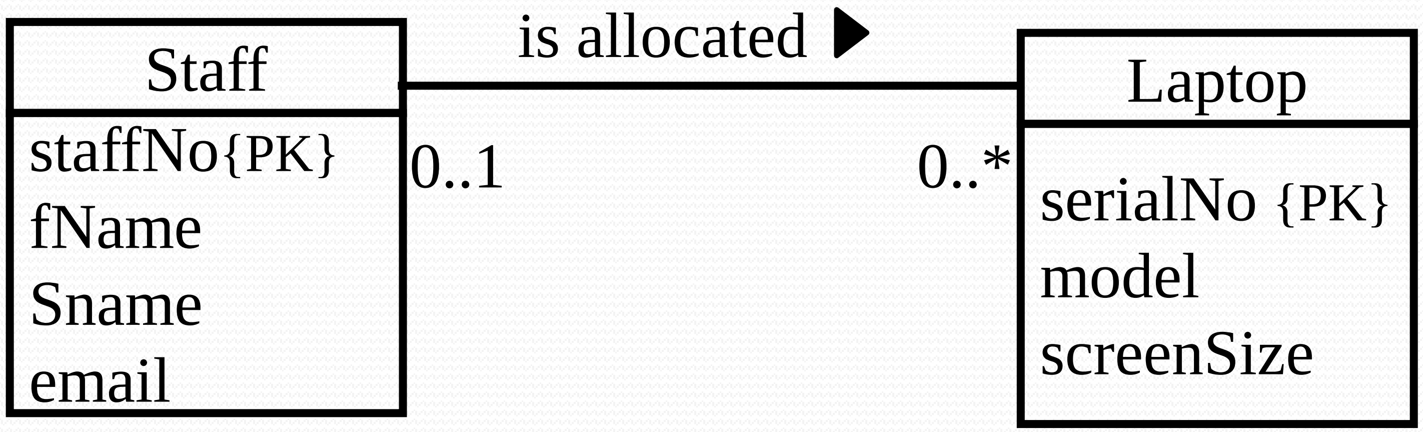
- **Primary Key**
  - Candidate key selected to identify tuples uniquely within relation.
- **Alternate Keys**
  - Candidate keys that are not selected to be primary key.
- **Foreign Key**
  - Attribute, or set of attributes, within one relation that matches candidate key of some (possibly same) relation.

1) One-to-many (1:M) relationship

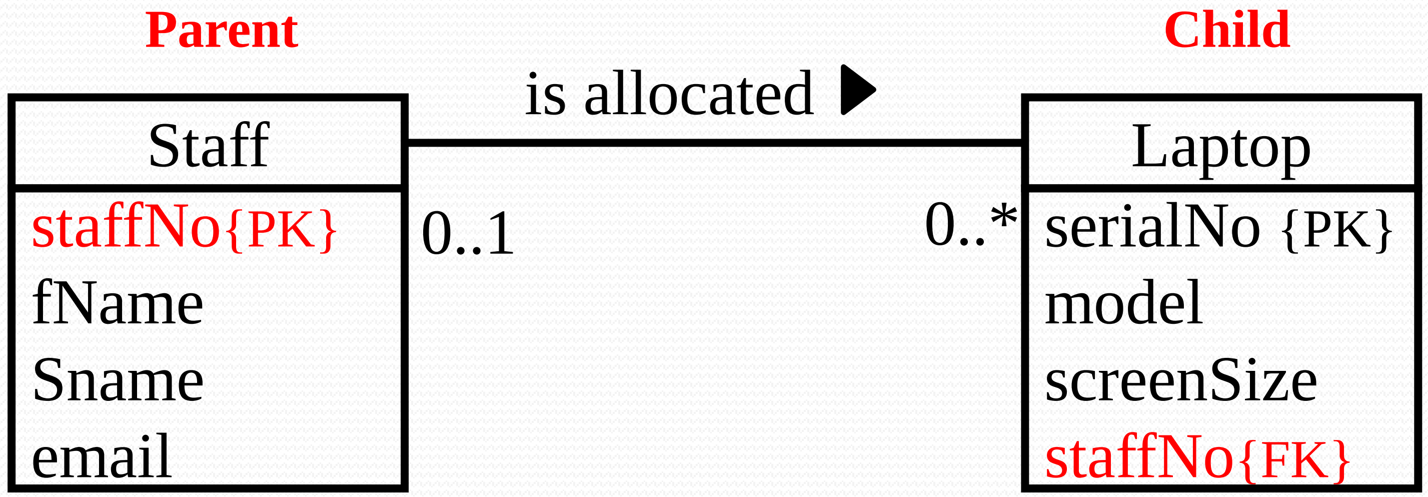
- Create 2 tables
- Parent table on the “one” side
- Child table on the “many” side
- Create FK on the Child table as a copy of the PK of the Parent table
- FK of the Child Table references the PK of Parent Table

1) One-to-many (1:M) relationship

Conceptual



Logical



Tables

Staff (staffNo{PK}, fName, sName, email)

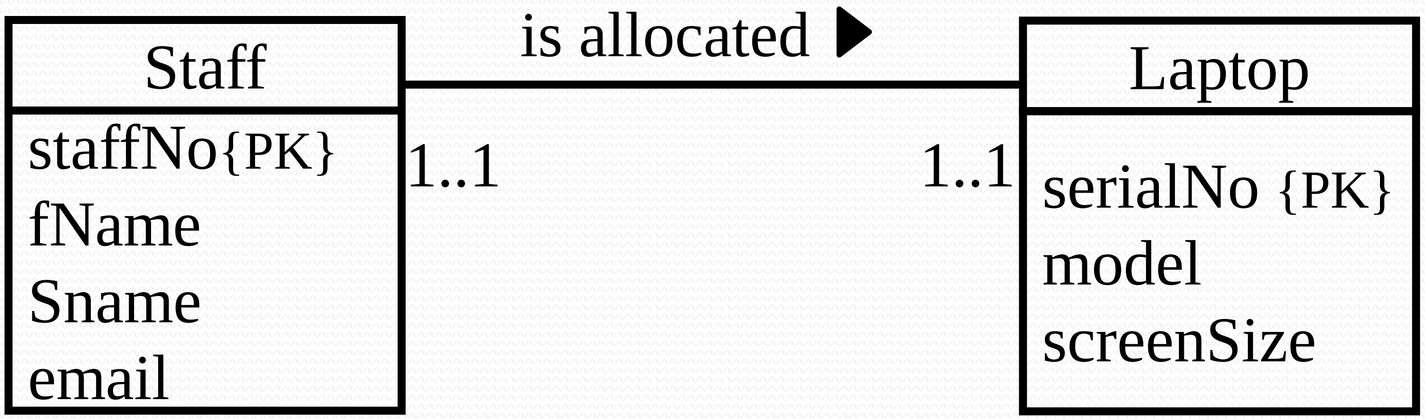
Laptop (serialNo {PK}, model, screenSize, staffNo{FK})

2) One-to-one (1:1) mandatory on both sides

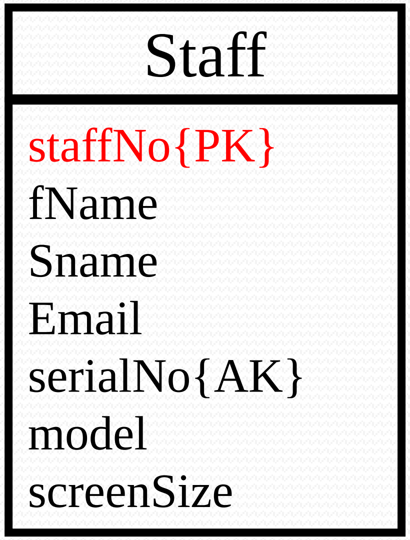
- Create ONE table
- Merge 2 tables into one
- Choose one PK from the two PKs, the other one is AK

2) One-to-one (1:1) mandatory on both sides

Conceptual



Logical



Tables

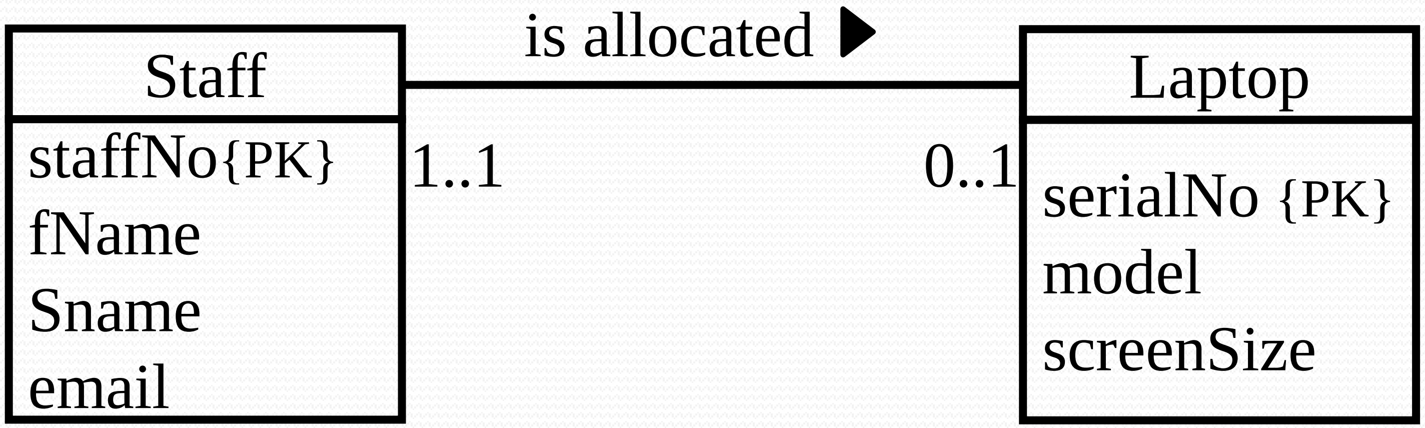
Staff (staffNo{PK}, fName, sName, email, serialNo{AK}, model, screenSize)

3) One-to-one (1:1) optional on one side

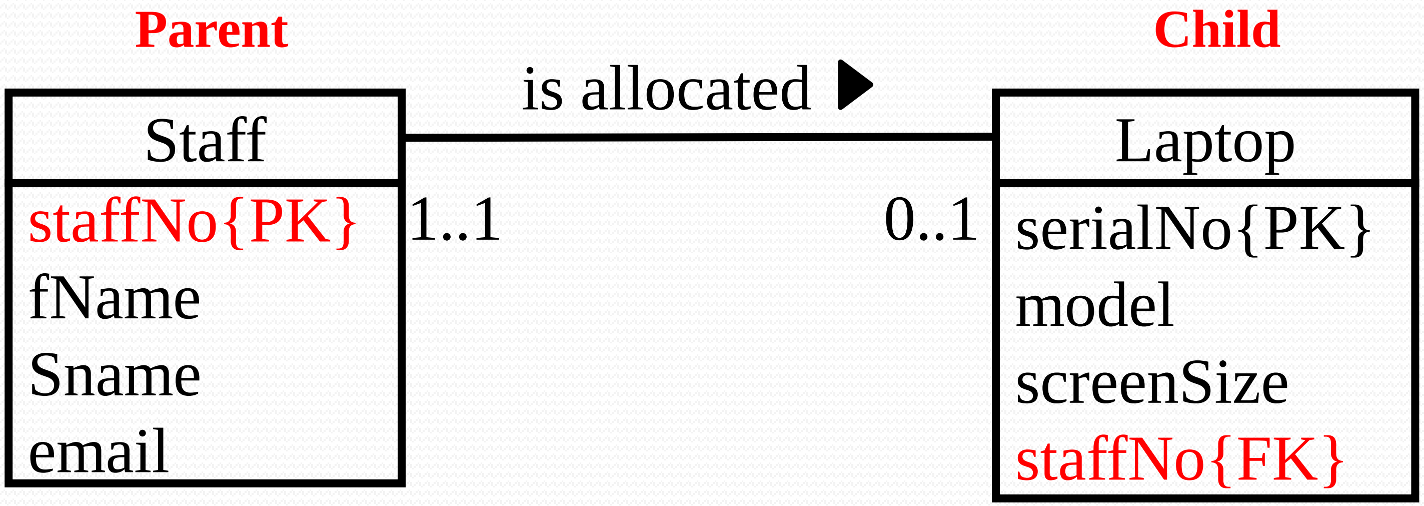
- Create TWO tables
- Parent table on “mandatory” side
- Child table on the “optional” side
- Create FK on the Child table as a copy of the PK of the Parent table
- FK of the Child table references the PK of Parent Table

3) One-to-one (1:1) optional on one side

Conceptual



Logical



Tables

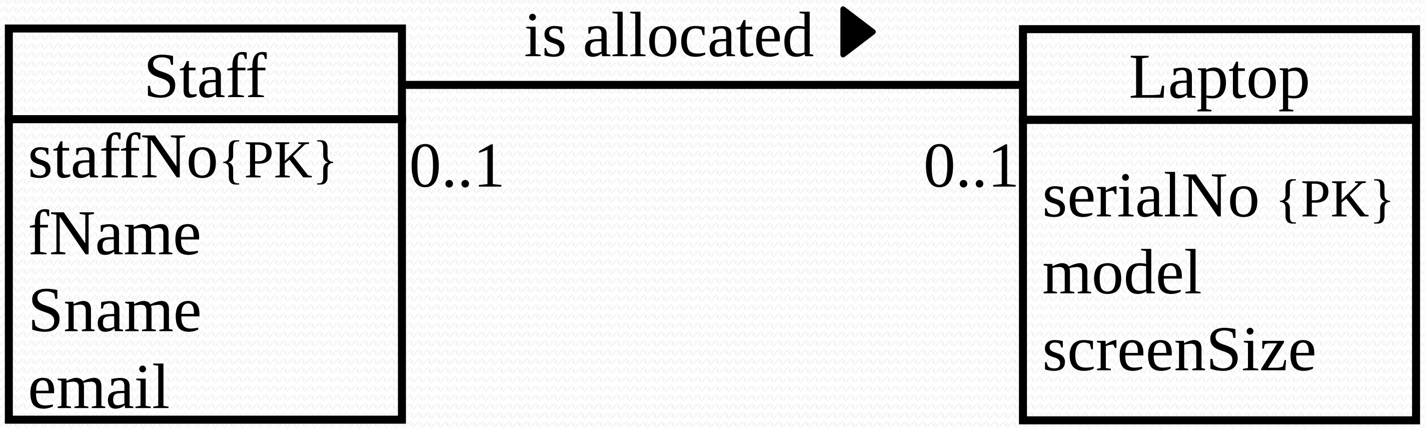
Staff (staffNo{PK}, fName, sName, email)
Laptop (serialNo{PK}, model, screenSize, staffNo{FK})

4) One-to-one (1:1) optional on both sides

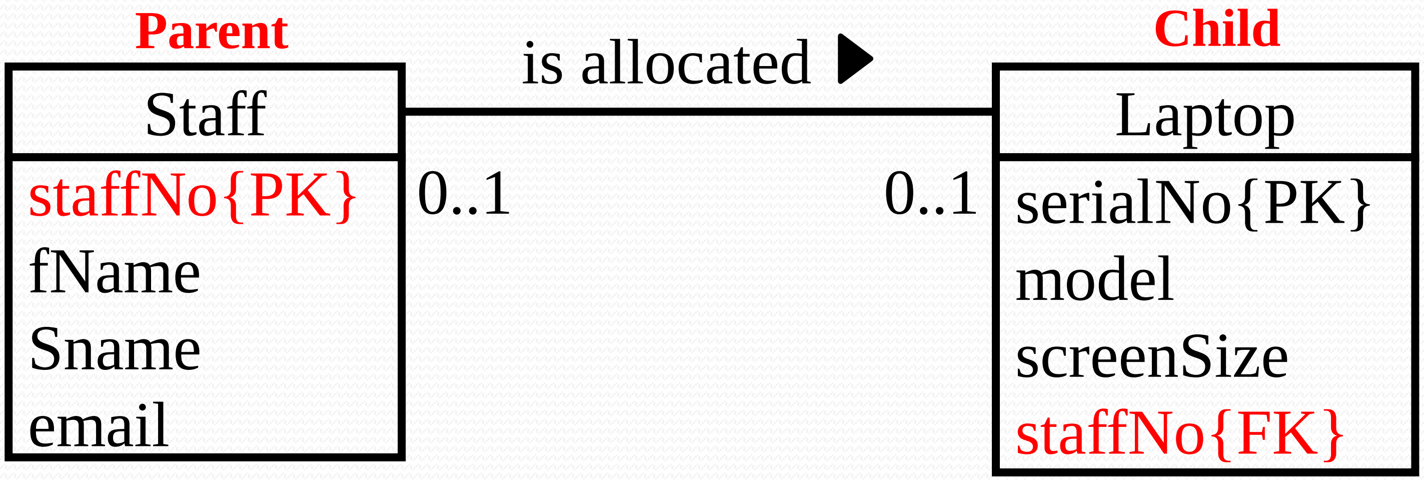
- Create TWO tables
- If info available, Parent table on “more mandatory” side and Child table on the “more optional” side
- If no info available, choose the Parent & the Child Table
- Create FK on the Child table as a copy of the PK of the Parent table
- FK of the Child Table references the PK of Parent Table

4) One-to-one (1:1) optional on both sides

Conceptual



Logical



Tables

Staff (staffNo{PK}, fName, sName, email)
Laptop (serialNo{PK}, model, screenSize, staffNo{FK})

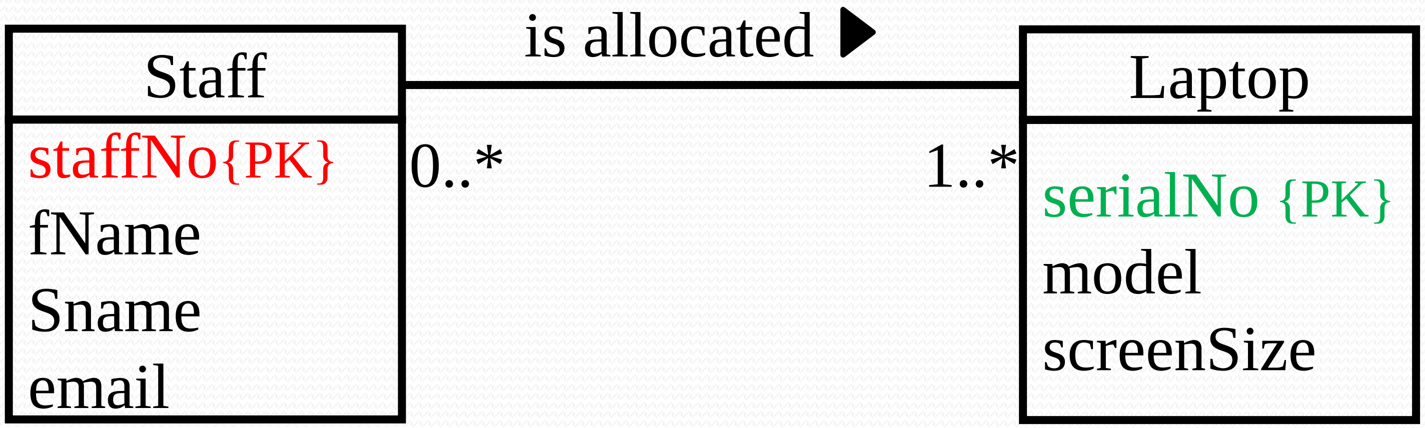
5) Many-to-Many

- Create THREE Tables
- 2 original Parent tables
- 1 Link table associated to the two Parent tables through two 1:M relationships

  → Link table is the Child table of both Parent tables
- FKs of Link Child table reference the PKs of the Parent tables
- PK of Link Child table combination of the 2 PKs of the Parent Tables
- Compound PK if combination of 2 PKs will never be repeated
- Composite PK with additional date (and possibly time) if combination of 2 PKs can be repeated

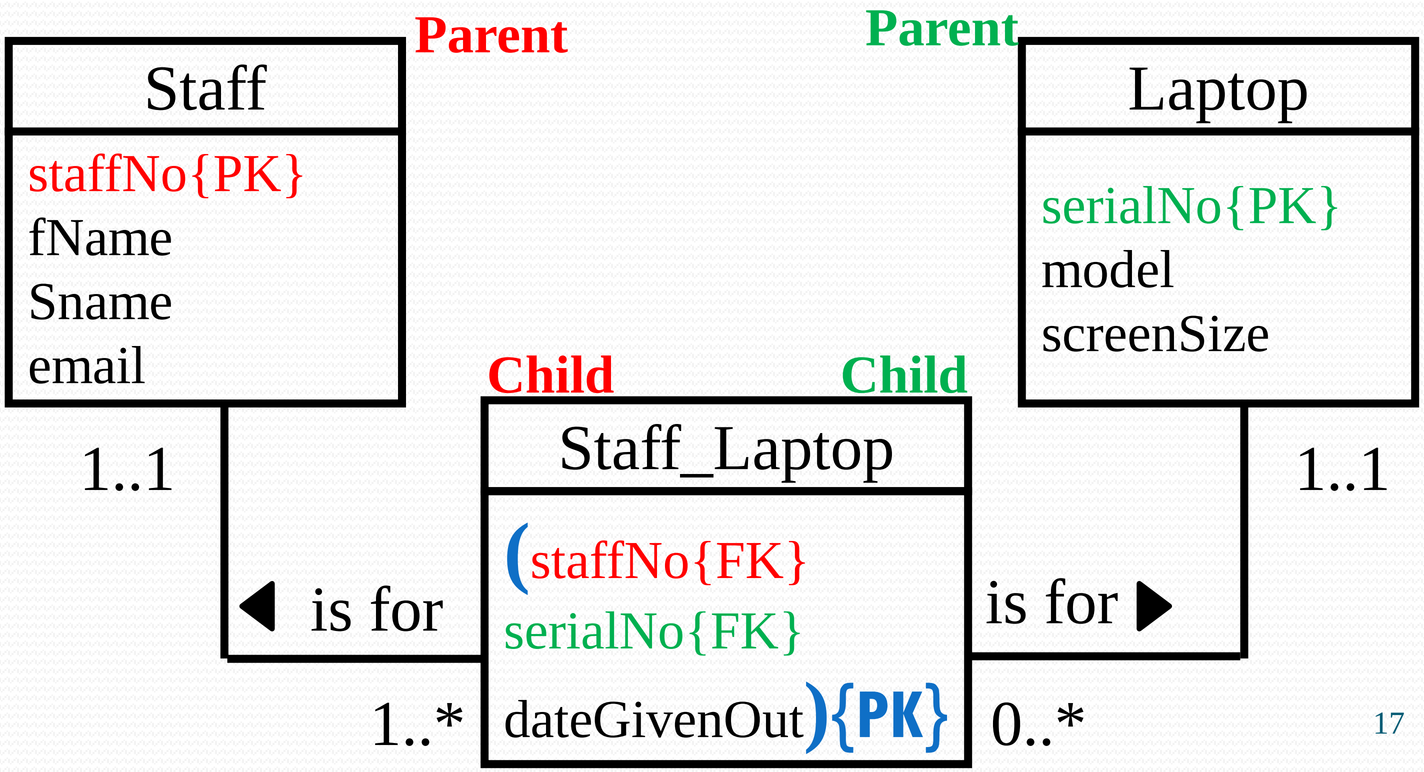
5) Many-to-Many

Conceptual



Logical

THINK! Compound or Composite PK?



5) Many-to-Many (continued)

Tables	Staff (staffNo{PK}, fName, sName, email) Laptop (serialNo{PK}, model, screenSize) Staff_Laptop ((staffNo{FK}, serialNo{FK}, dateGivenOut){**PK**})

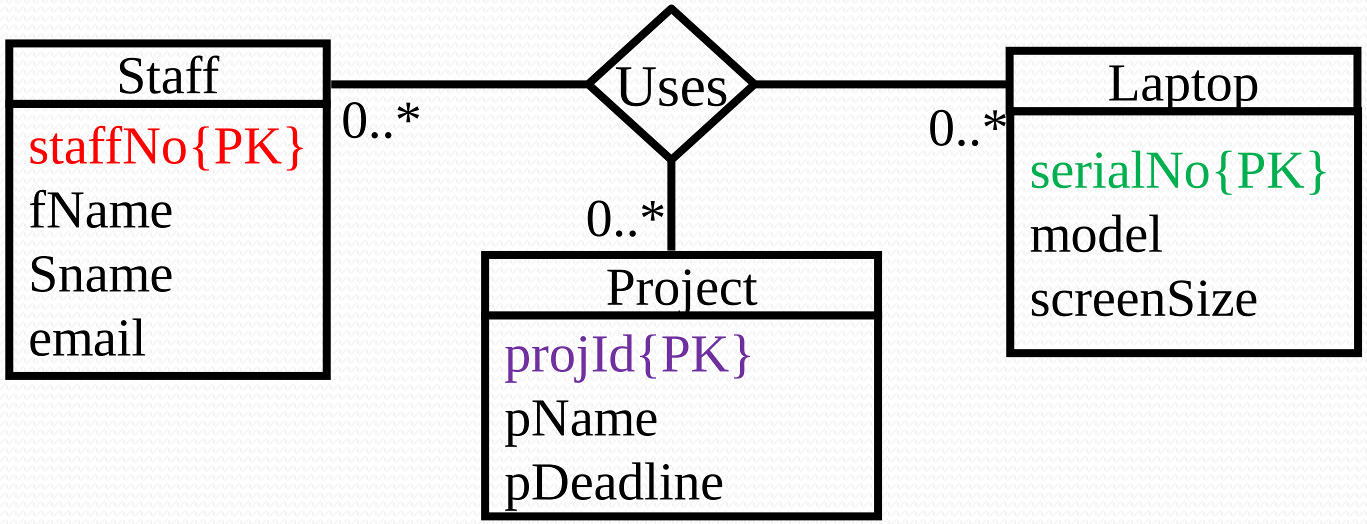
6) Complex relationships: ternary & quaternary

- For ternary, create FOUR Tables
- 3 original Parent tables
- 1 Link table associated to the two Parent tables through three 1:M relationships

  → Link table is the Child table of all 3 Parent tables
- FKs of Link Child table reference the PKs of the Parent tables
- PK of Link Child table combination of the 3 PKs of the Parent Tables
- Compound PK if combination of 3PKs will never be repeated
- Composite PK with additional date (and possibly time) if combination of 3 PKs can be repeated

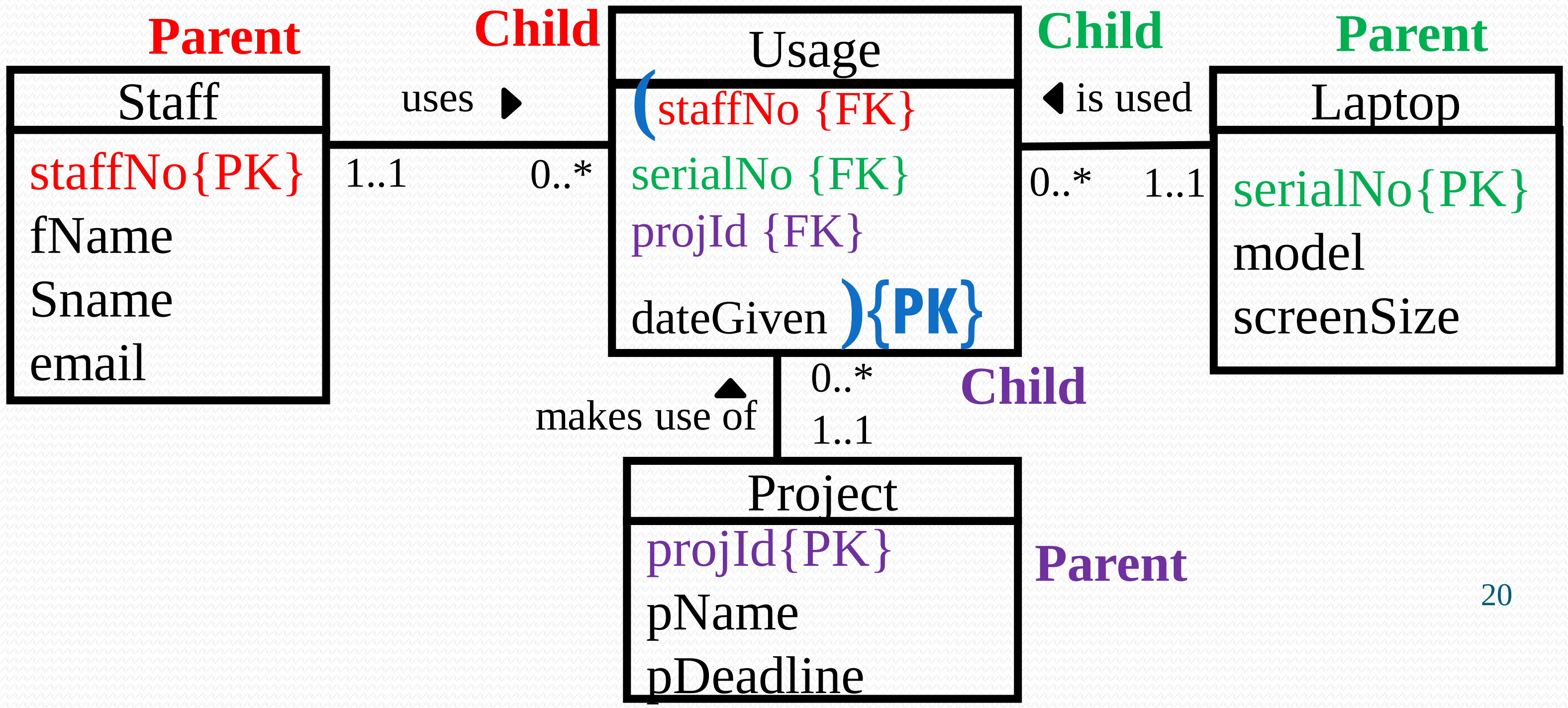
6) Complex relationships: ternary & quaternary

Conceptual



Logical

THINK! Compound or Composite PK?



6) Complex relationships (contd)

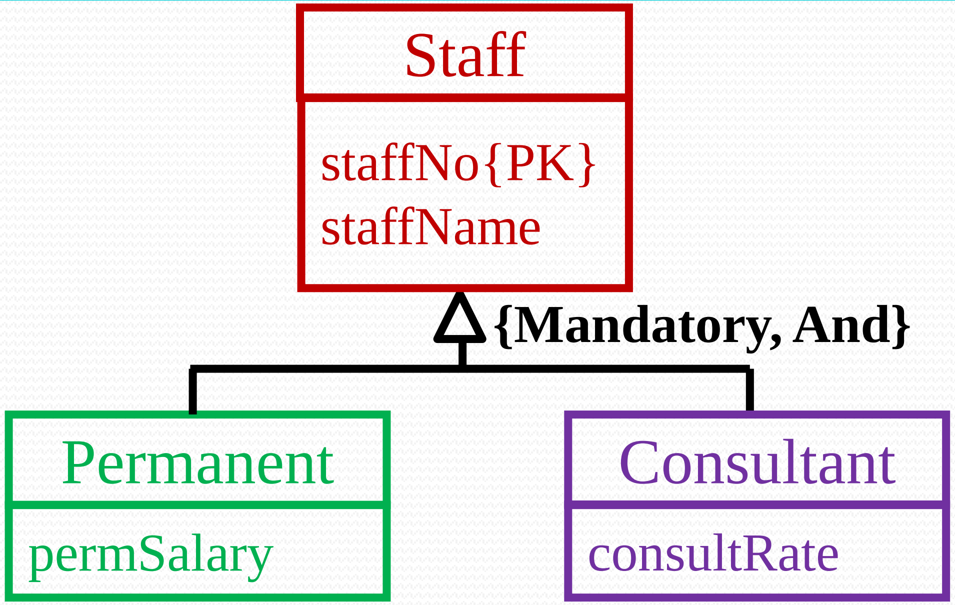
Tables	Staff (staffNo{PK}, fName, sName, email) Laptop (serialNo{PK}, model, screenSize) Project (projId{PK}, pName) Usage ((staffNo{FK}, serialNo{FK}, projId{FK}, dateGiven){PK})

7) Generalisation with {Mandatory, And}

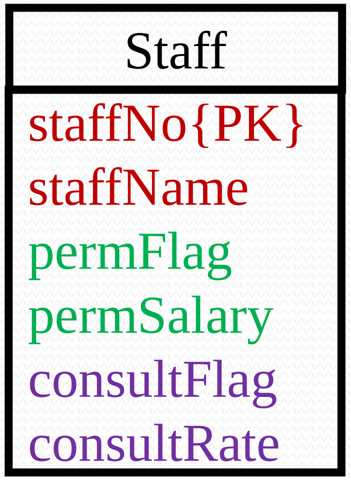
- Create ONE table
- New table combines attributes of all entities
- PK of new table: PK of general entity
- Use flags to differentiate between records of previous sub-entities.

7) Generalisation with {Mandatory, And}

Conceptual



Logical



Tables

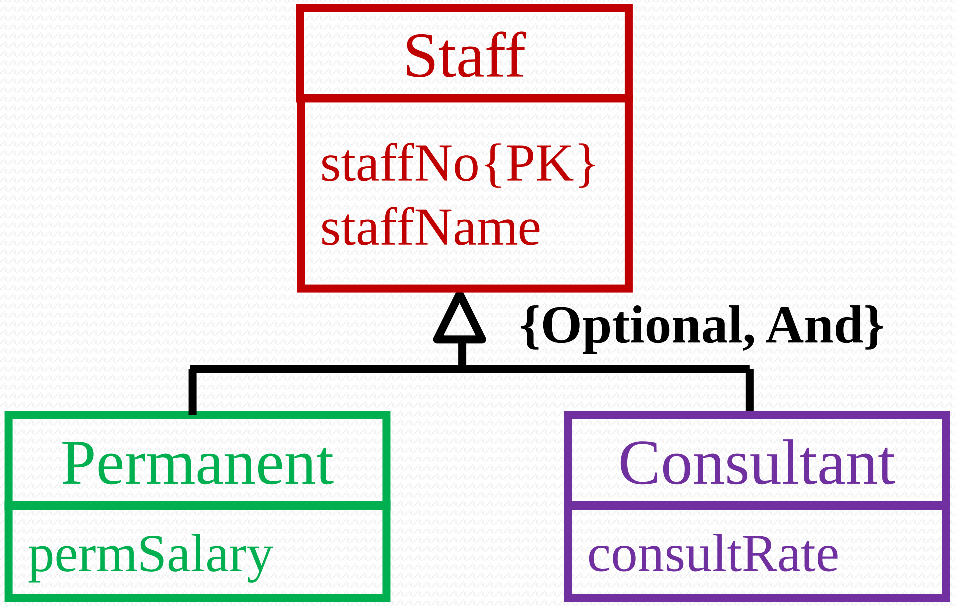
Staff (staffNo{PK}, staffName, permSalary, consultRate, permFlag, consultFlag)

8) Generalisation with {Optional, And}

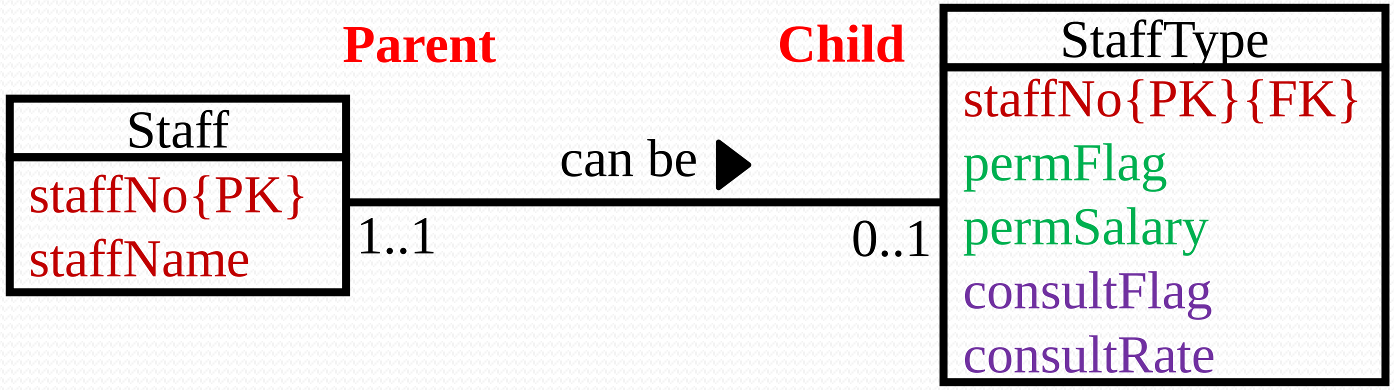
- Create TWO Tables and a 1:1 relationship optional on one side
- One table for super-entity which becomes the Parent table
- One table for both sub-entities merged together, which becomes the Child table
- PK of the Child table is the same as the PK of Parent table
- FK of the Child table references PK of the Parent table
- Use flags to differentiate between records of previous sub-entities.

8) Generalisation with {Optional, And}

Conceptual



Logical



Tables

Staff (staffNo{PK}, staffName)

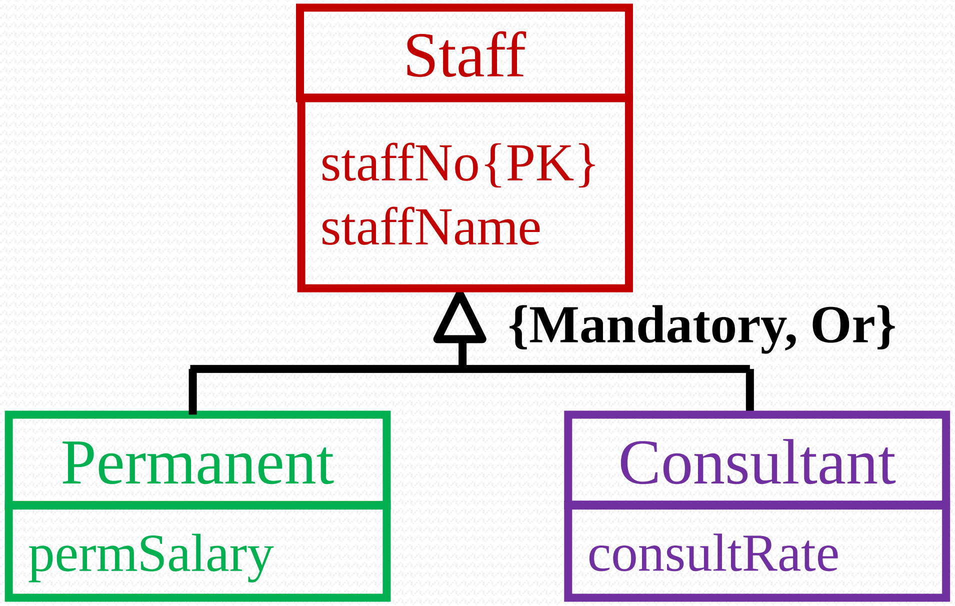
StaffDetails (staffNo{PK}{FK}, permFlag, permSalary, consultFlag, consultRate)

9) Generalisation with {Mandatory, Or}

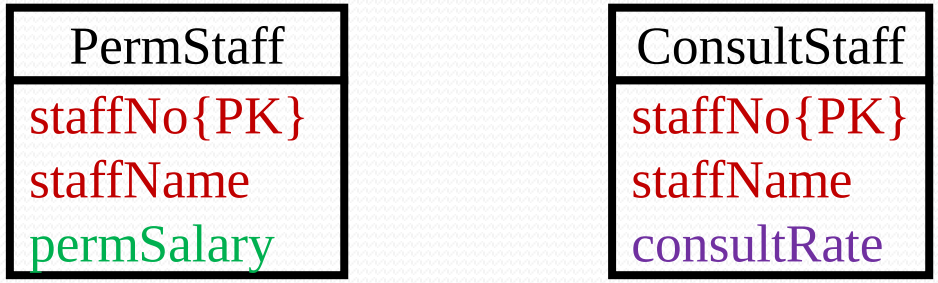
- Create TWO tables
- One table table for each of the sub-entities.
- PK for both tables is the PK of original super entity.
- Each table have their own relationships with the rest of the schema.

9) Generalisation with {Mandatory, Or}

Conceptual



Logical



Tables

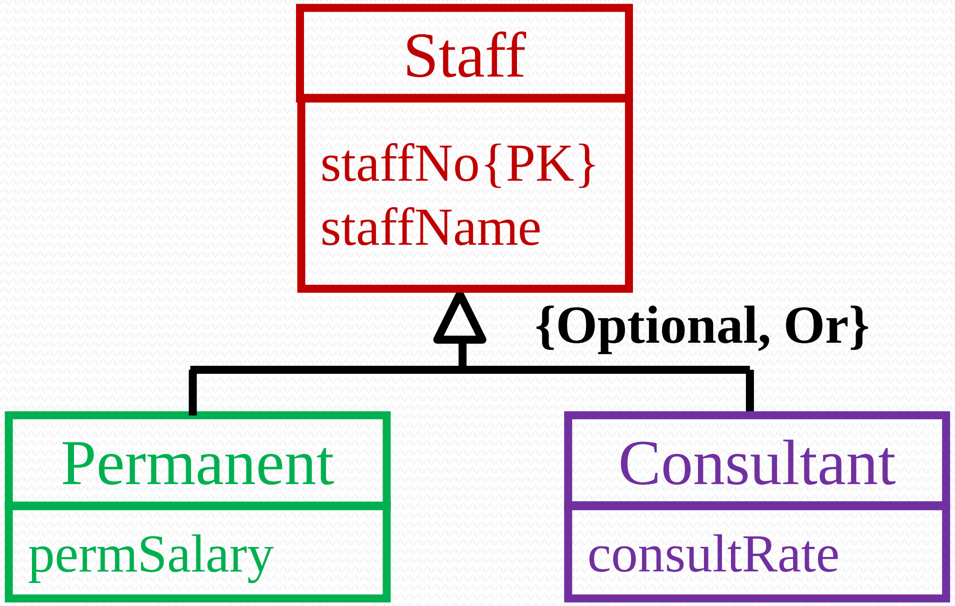
PermStaff (staffNo{PK}, staffName, permSalary)
ConsutlStaff(staffNo{PK}, staffName, consultRate)

10) Generalisation with {Optional, Or}

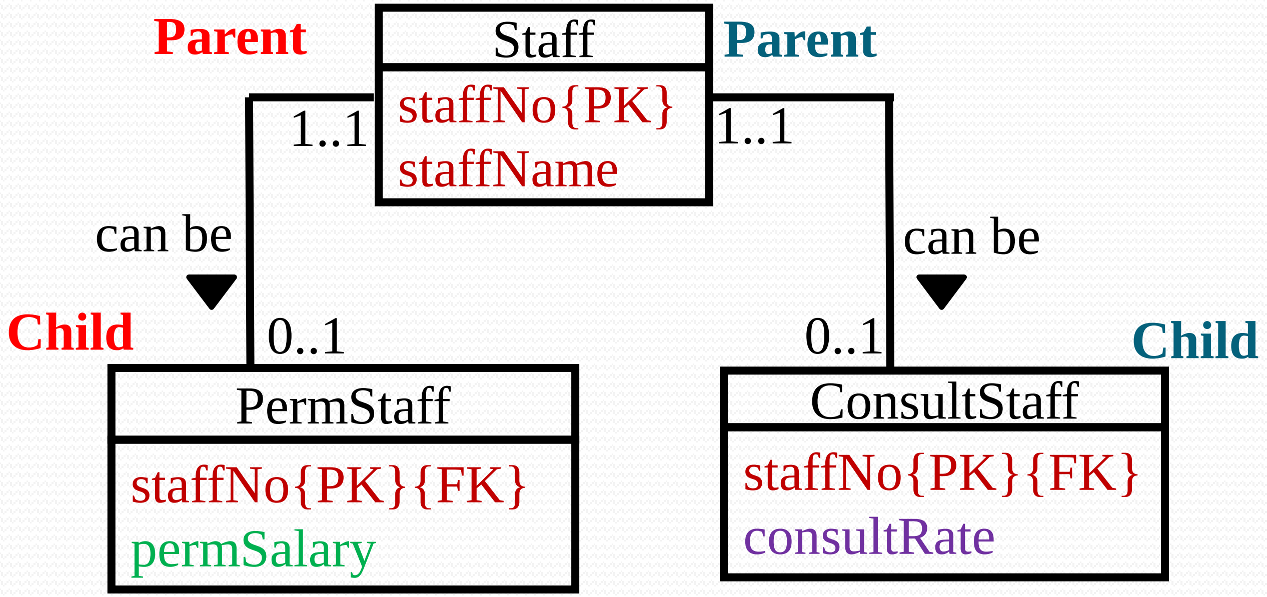
- Create THREE tables and two 1:1 relationships optional on one side
- One table for super-entity which becomes the Parent Table
- One table for each sub-entities, each becomes the Child table, for their respective relationships
- PK of the Child table is the same as the PK of Parent table
- FK of the Child table reference the PK of the Parent table

10) Generalisation with {Optional, Or}

Conceptual



Logical



10) Generalisation with {Optional, Or} (contd)

Tables	Staff (staffNo{PK}, staffName) PermStaff (staffNo{PK}{FK}, permSalary) ConsutlStaff(staffNo{PK}{FK},consultRate)