MONASH University

MONASH
INFORMATION
TECHNOLOGY

Database Design 1: Conceptual Modelling

*Please obtain a copy of the **Monash Software case study** from the week 2 page on Moodle under the "Pre-class activities" header*

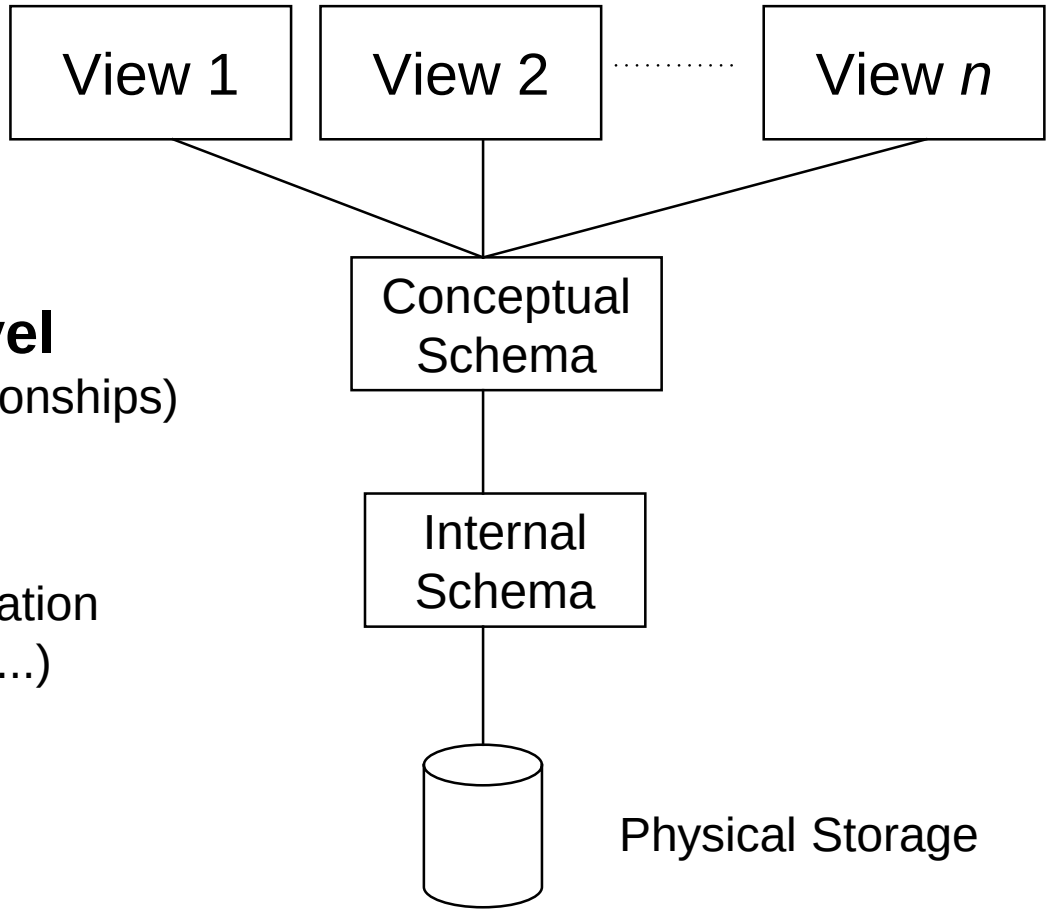
GROUP OF EIGHT AUSTRALIA

ANSI/SPARC architecture - proposed 1975

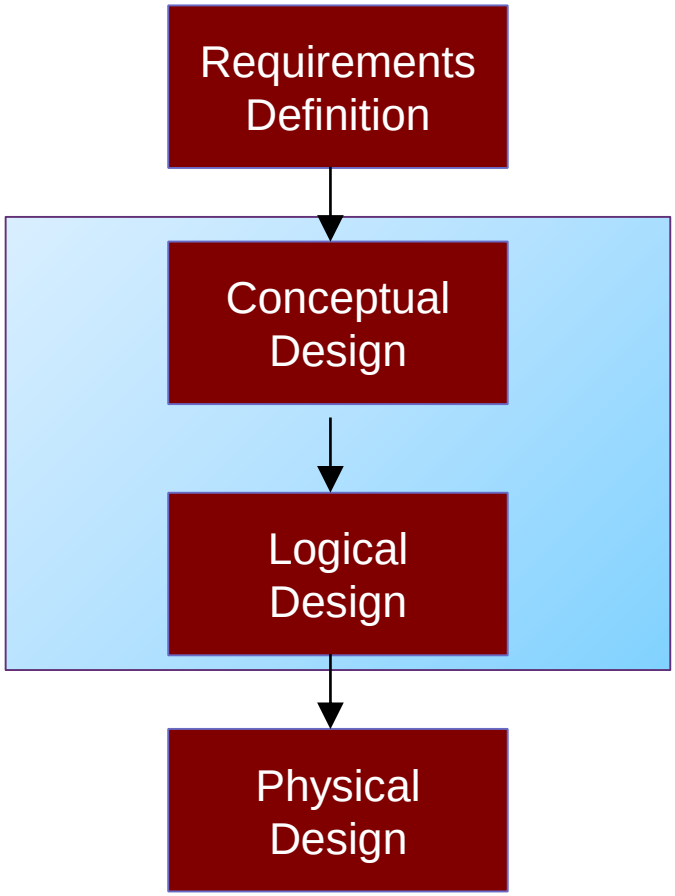
External level
(Users View)

Conceptual level
(All Data and Relationships)

Internal level
(Physical representation
- hash, btree, isam ...)



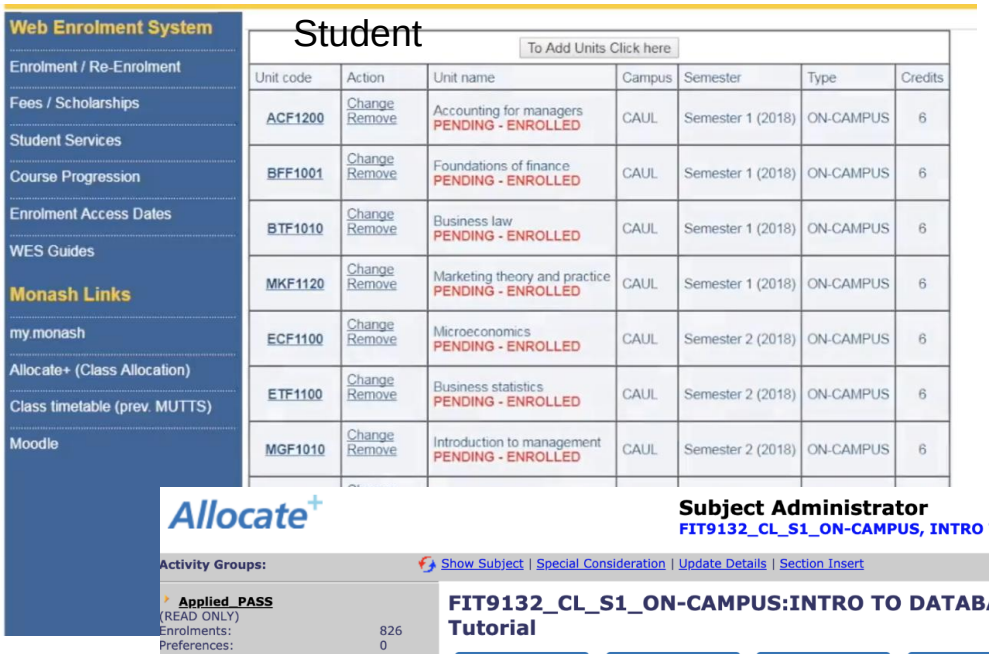
The Database Design Life Cycle



Requirements Definition

- Identify and analyse user views.
- A 'user view' may be a report to be produced or a particular type of transaction that should be supported.
- Corresponds to the external level of the ANSI/SPARC architecture.
- Output is a statement of specifications which describes the user views' particular requirements and constraints.

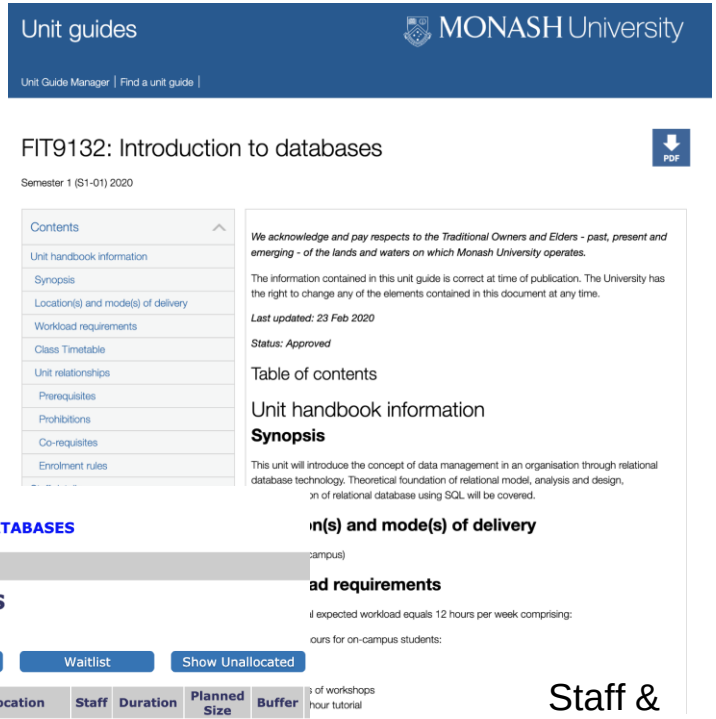
Different views of the underlying data

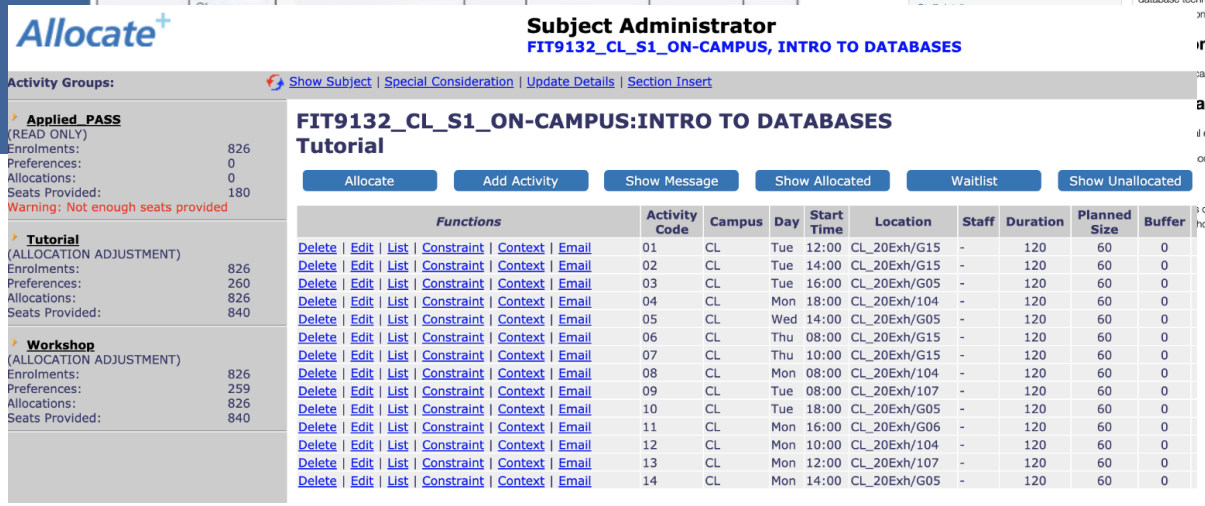


Student

To Add Units Click here

Unit code	Action	Unit name	Campus	Semester	Type	Credits
ACF1200	Change Remove	Accounting for managers PENDING - ENROLLED	CAUL	Semester 1 (2018)	ON-CAMPUS	6
BFF1001	Change Remove	Foundations of finance PENDING - ENROLLED	CAUL	Semester 1 (2018)	ON-CAMPUS	6
BTF1010	Change Remove	Business law PENDING - ENROLLED	CAUL	Semester 1 (2018)	ON-CAMPUS	6
MKF1120	Change Remove	Marketing theory and practice PENDING - ENROLLED	CAUL	Semester 1 (2018)	ON-CAMPUS	6
ECF1100	Change Remove	Microeconomics PENDING - ENROLLED	CAUL	Semester 2 (2018)	ON-CAMPUS	6
ETF1100	Change Remove	Business statistics PENDING - ENROLLED	CAUL	Semester 2 (2018)	ON-CAMPUS	6
MGF1010	Change Remove	Introduction to management PENDING - ENROLLED	CAUL	Semester 2 (2018)	ON-CAMPUS	6





Functions	Activity Code	Campus	Day	Start Time	Location	Staff	Duration	Planned Size	Buffer
Delete | Edit | List | Constraint | Context | Email	01	CL	Tue	12:00	CL_20Exh/G15	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	02	CL	Tue	14:00	CL_20Exh/G15	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	03	CL	Tue	16:00	CL_20Exh/G05	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	04	CL	Mon	18:00	CL_20Exh/104	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	05	CL	Wed	14:00	CL_20Exh/G05	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	06	CL	Thu	08:00	CL_20Exh/G15	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	07	CL	Thu	10:00	CL_20Exh/G15	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	08	CL	Mon	08:00	CL_20Exh/104	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	09	CL	Tue	08:00	CL_20Exh/107	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	10	CL	Tue	18:00	CL_20Exh/G05	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	11	CL	Mon	16:00	CL_20Exh/G06	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	12	CL	Mon	10:00	CL_20Exh/104	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	13	CL	Mon	12:00	CL_20Exh/107	-	120	60	0
Delete | Edit | List | Constraint | Context | Email	14	CL	Mon	14:00	CL_20Exh/G05	-	120	60	0

Staff & Student

Admin

ER Modeling

- ER (Entity-Relationship) model developed by Peter Chen in 1976 to aid database design.
- Used for conceptual model (ERD).
- ER diagrams give a visual indication of the design.
- Basic components:
  - Entity
  - Attribute
  - Relationship

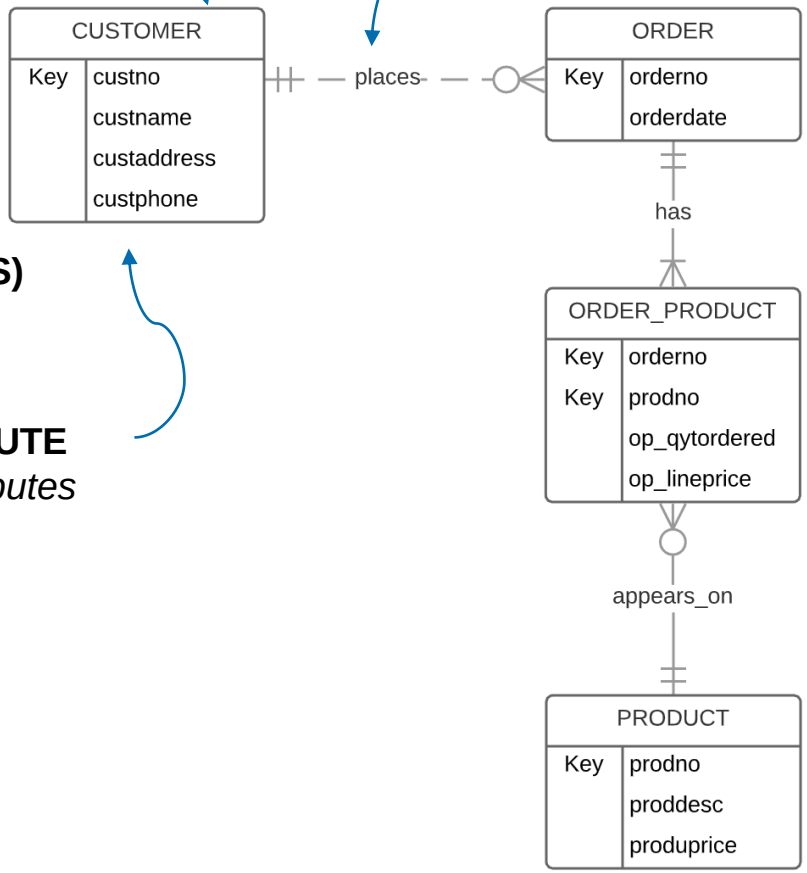
Conceptual Design

- Develop the enterprise data model.
- Corresponds to the conceptual level of the ANSI/SPARC architecture.
- *Independent of all physical implementation considerations (the type of database to be used)*.
- Various design methodologies may be employed such as UML, ER (Entity-Relationship).
- ER consists of ENTITIES and RELATIONSHIPS between entities
  - –An ENTITY will have attributes (things we wish to record), one or more of which will identify an entity instance (called the KEY)

Conceptual Level (ER Model)

ENTITY
Collection of "Customer(s)"

RELATIONSHIP
Relates entities



KEY ATTRIBUTE(S)
Instance identifier

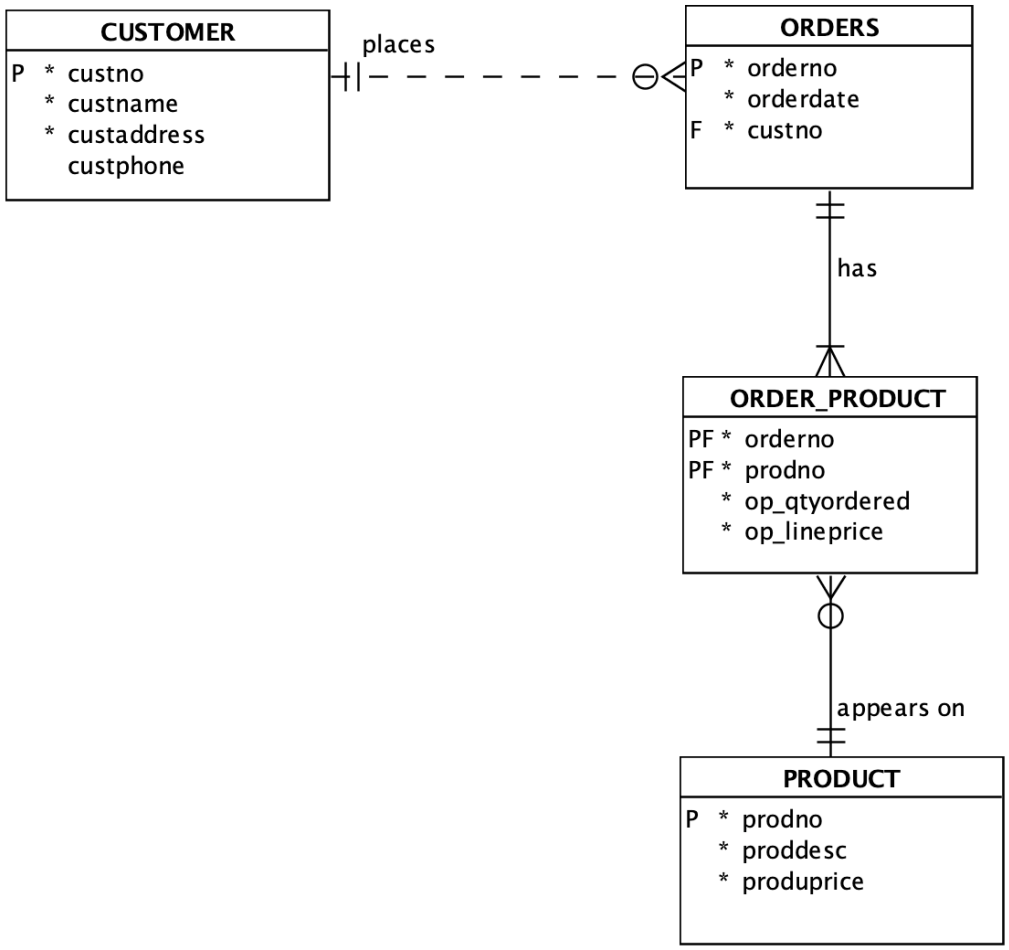
NON KEY ATTRIBUTE
Other non-key attributes

Assignment 1A

Logical Design

- Develop a data model which targets a particular database model (e.g. relational, hierarchical, network, object-oriented, noSQL).
- Independent of any implementation details which are specific to any particular vendors DBMS package.
- Normalisation technique (see week 4) is used to test the correctness of a relational logical model.

Logical Level (Logical Model - Relational)



Assignment 1B

Physical Design

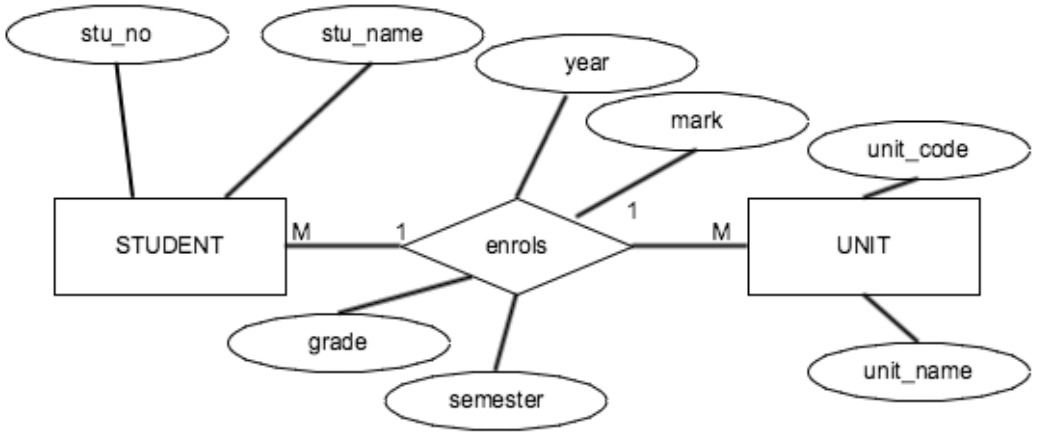
- Develop a strategy for the physical implementation of the logical data model.
- Choose appropriate storage structures, indexes, file organisations and access methods which will most efficiently support the user requirements (not part of unit).
- Physical design phase is dependent on the particular DBMS environment in use.
- ANSI/SPARC internal level.

Physical Level – Starting point

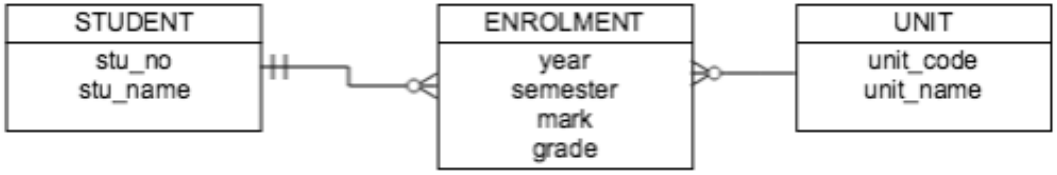
Oracle Database 12c | Relational_1 | Generate

```
 8  CREATE TABLE customer (
 9      custno        NUMBER(7) NOT NULL,
10      custname      VARCHAR2(50) NOT NULL,
11      custaddress   VARCHAR2(50) NOT NULL,
12      custphone     CHAR(10)
13  );
14
15  COMMENT ON COLUMN customer.custno IS
16      'Customer number';
17
18  COMMENT ON COLUMN customer.custname IS
19      'Customer name';
20
21  COMMENT ON COLUMN customer.custaddress IS
22      'Customer address';
23
24  COMMENT ON COLUMN customer.custphone IS
25      'Customer phone number';
26
27  ALTER TABLE customer ADD CONSTRAINT customer_pk PRIMARY KEY ( custno );
28
29  CREATE TABLE order_product (
30      orderno         NUMBER(7) NOT NULL,
31      prodno          NUMBER(7) NOT NULL,
32      op_qtyordered   NUMBER(3) NOT NULL,
33      op_lineprice    NUMBER(8, 2) NOT NULL
34  );
35
```

ERD - Notation

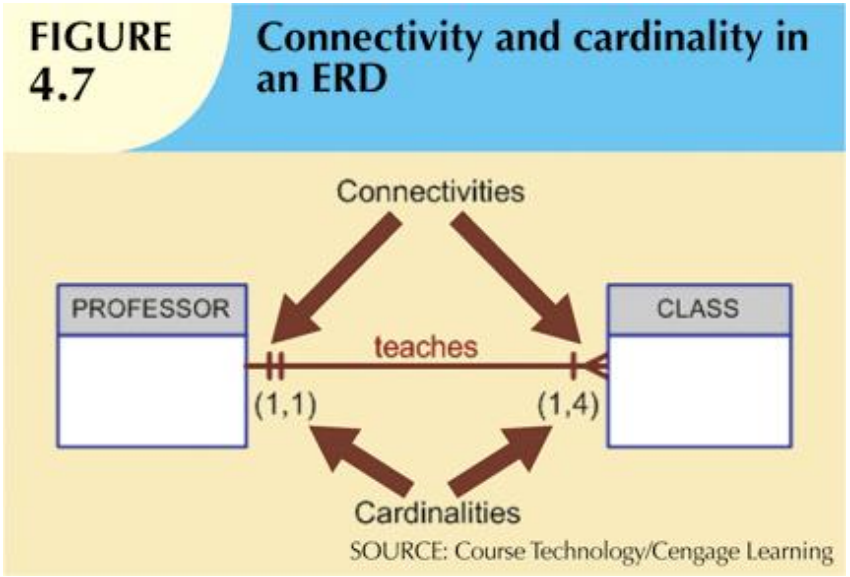


Chen



Information Engineering/James Martin/Crows foot

**** This is what we will be using***

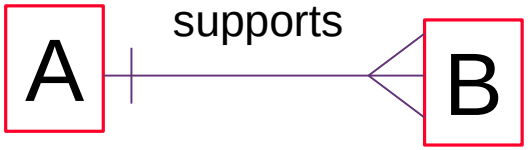


In general for Crows Foot notation specific cardinalities are not shown as above eg. (1,4), instead participation is depicted via min and max participation using the standard symbols (Inside symbol = min, outside symbol = max)

CONNECTIVITY

one to one

one to many



many to many

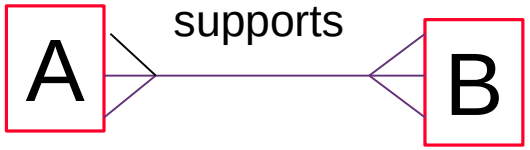
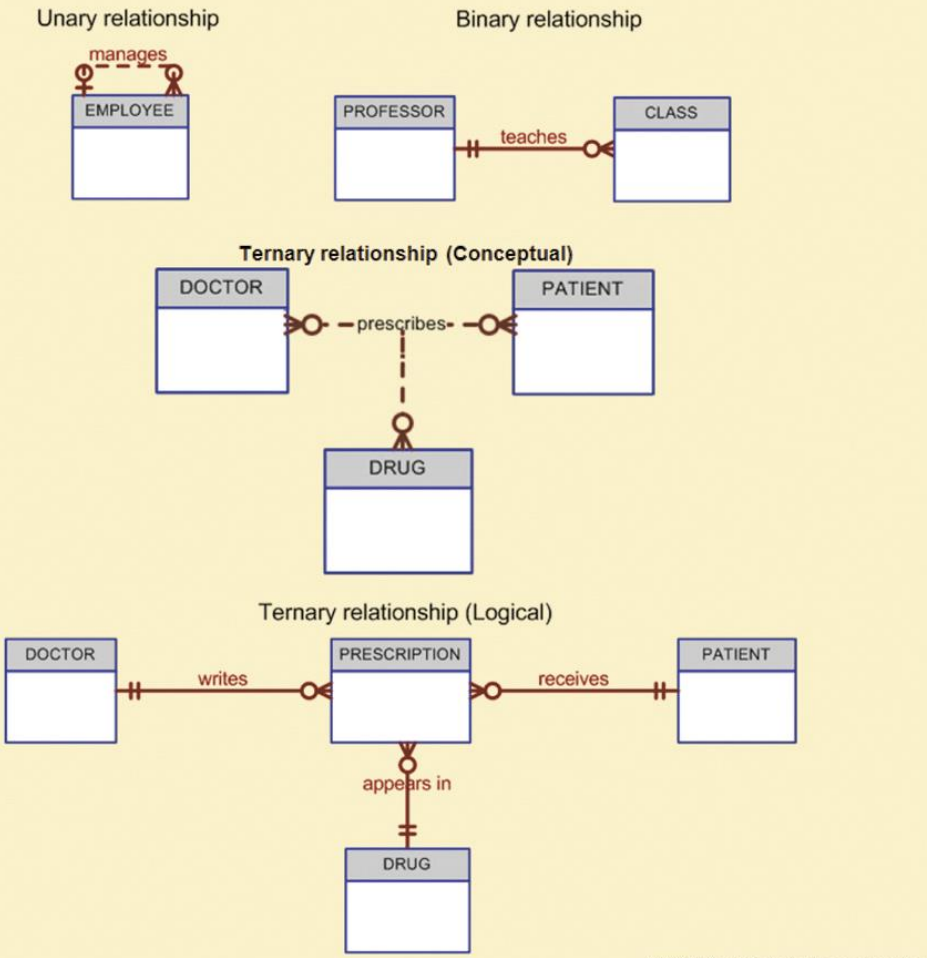


FIGURE 4.15 Three types of relationship degree

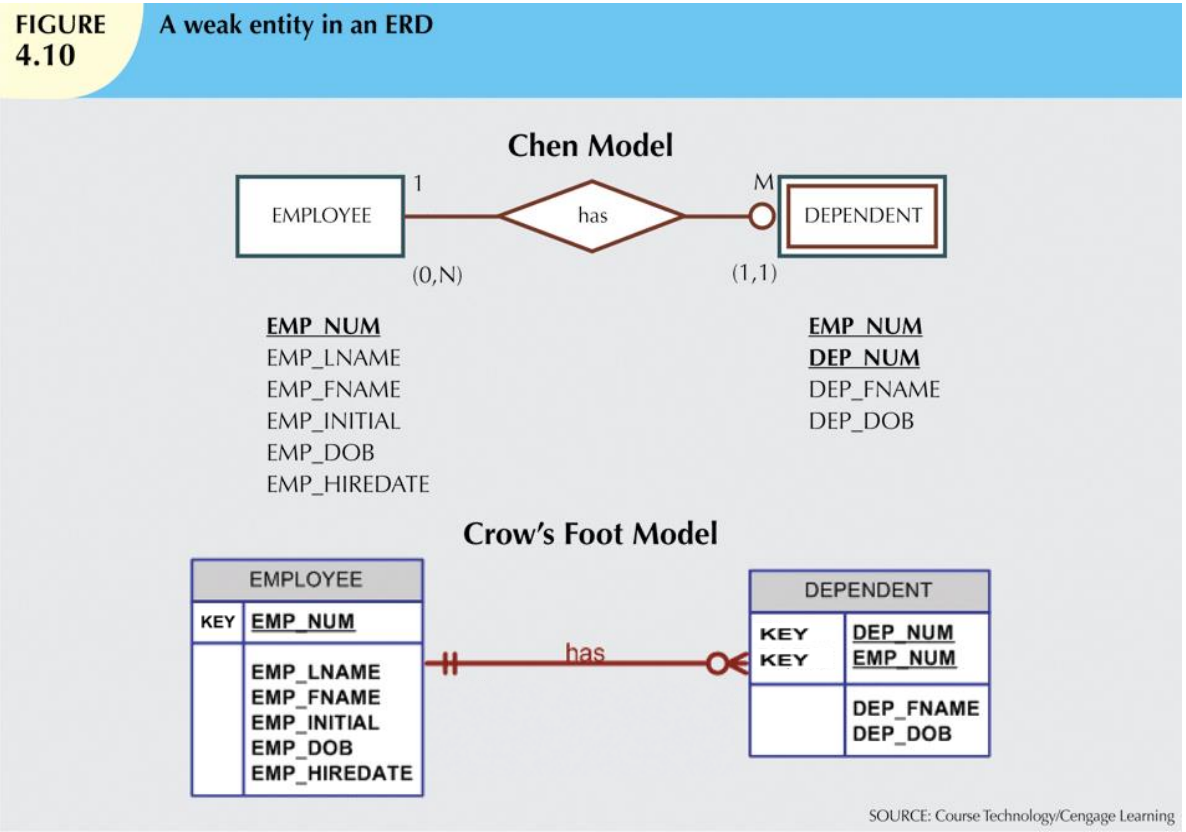


SOURCE: Course Technology/Cengage Learning

Weak vs Strong Entity

- Strong entity
  - Has a key which may be defined without reference to other entities.
  - For example EMPLOYEE entity.
- Weak entity
  - Has a key which requires the existence of one or more other entities.
  - For example FAMILY entity - need to include the key of employee to create a suitable key for family
- Database designer often determines whether an entity can be described as weak based on business rules
  - customer pays monthly account
    - Key: cust_no, date_paid, or
    - Key: payment_no (surrogate? – not at conceptual level)

Weak vs Strong Entity



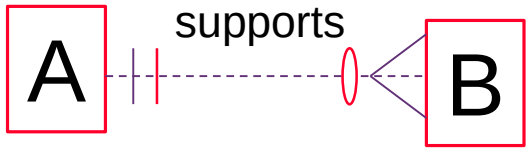
Note the Crow's Foot model shown here has been modified from the text version

Identifying vs Non-Identifying Relationship

- **Identifying**
  - Identifier of A is part of identifier of B.

  - Shown with solid line
  - ENROLMENT - STUDENT Enrolment key includes student id, which is an identifier of student.

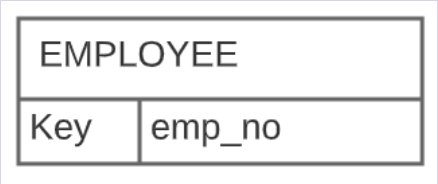
- **Non-identifying**
  - Identifier of A is NOT part of identifier of B.



  - Shown with broken line
  - Department no (identifier of department) is not part of Employee's identifier.

Student Activities

- Using the Monash Software Case Study, identify the entities present and their key attribute/s
  - draw a box for each entity, name it and add the key attribute/s
  - For example:



Entities in the Monash Software Case Study

TRAINING	
Key	training_code

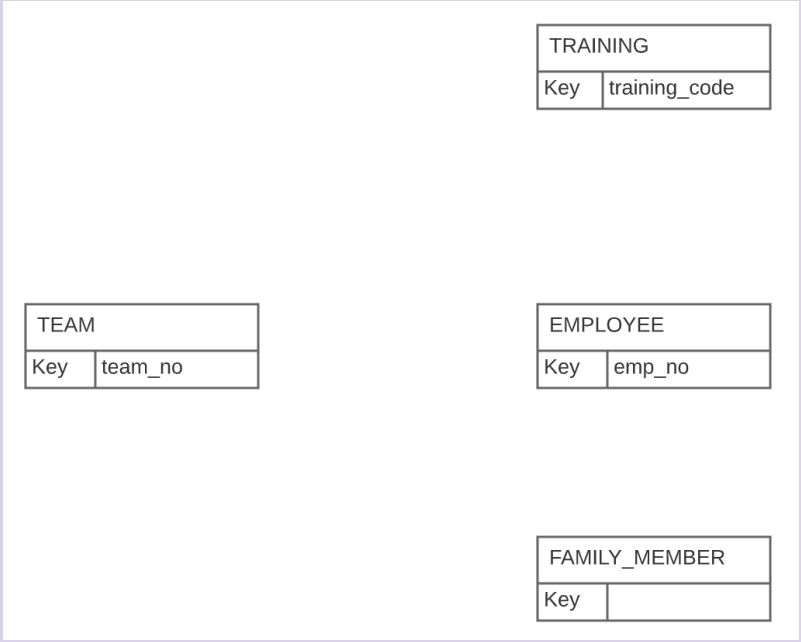
TEAM	
Key	team_no

EMPLOYEE	
Key	emp_no

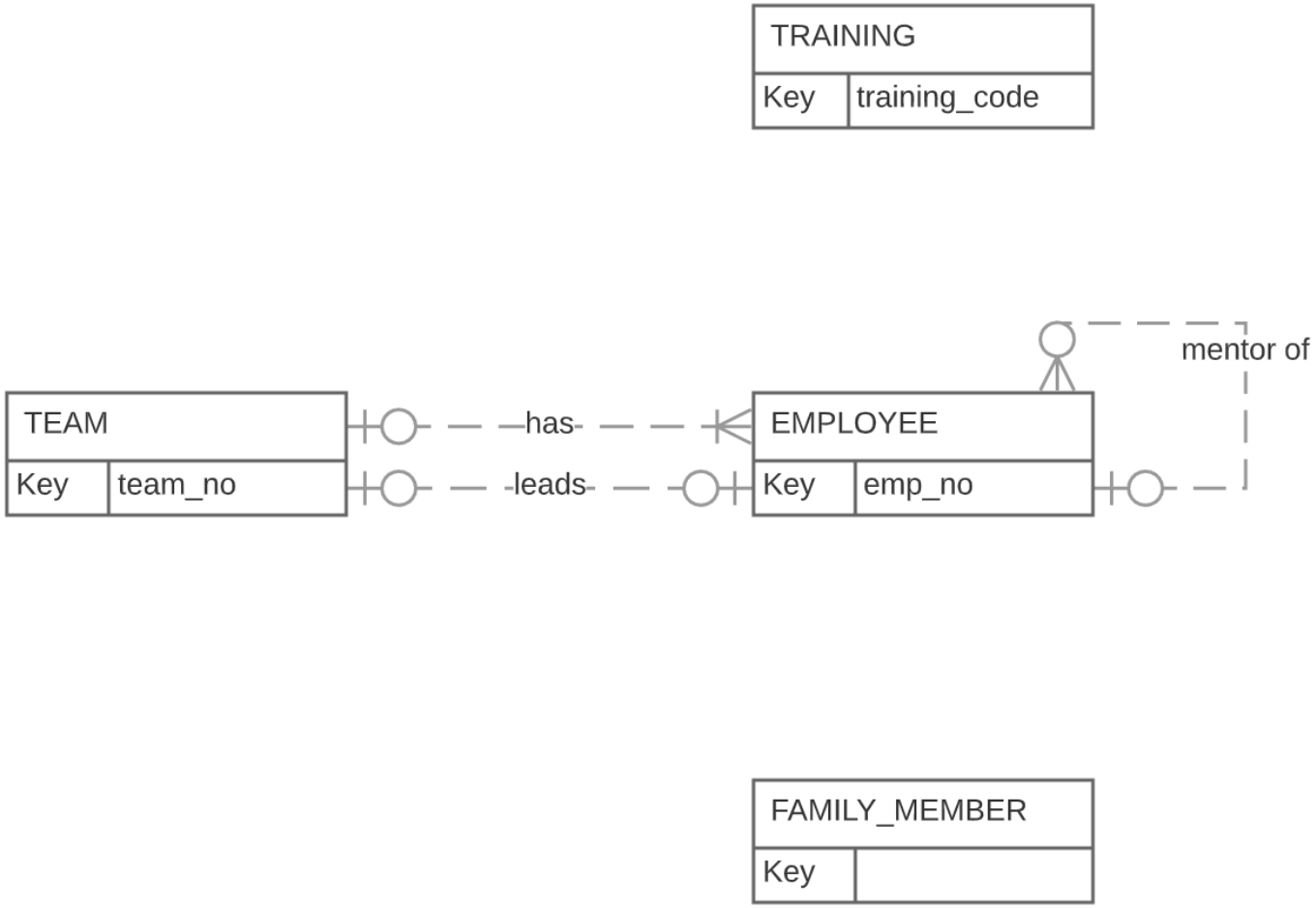
FAMILY_MEMBER	
Key	

Student Activities

- Using the Monash Software Case Study, identify the relationship(s) and participation which exist between TEAM and EMPLOYEE



TEAM - EMPLOYEE relationships

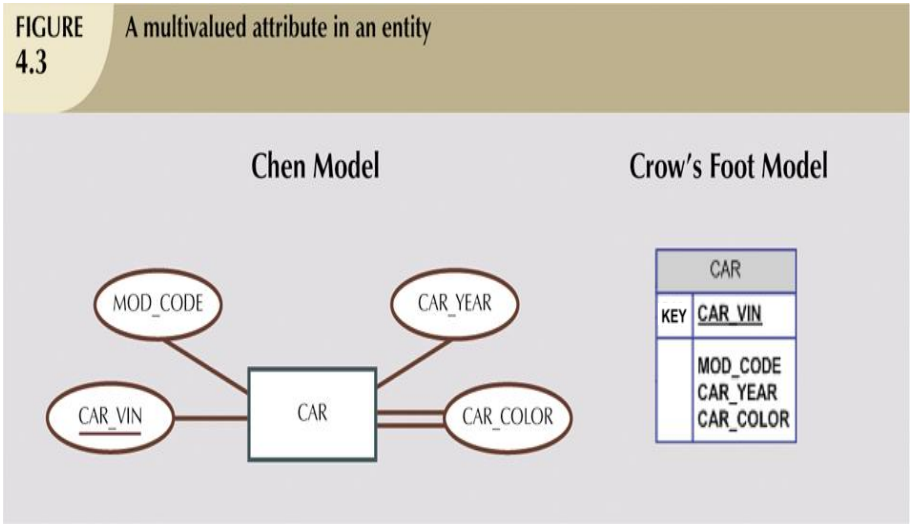


Types of Attributes

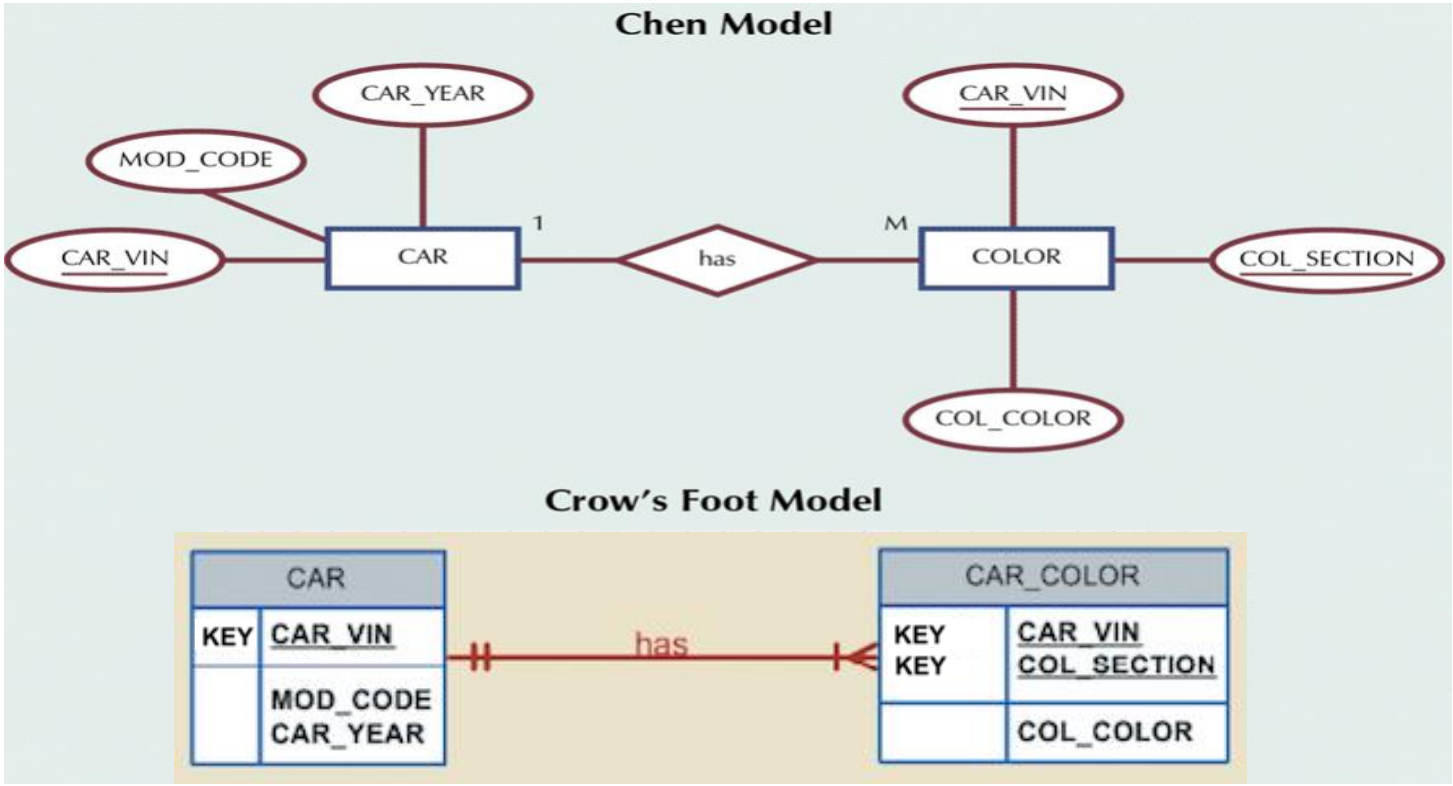
- Simple
  - Cannot be subdivided
  - Age, sex, marital status
- Composite
  - Can be subdivided into additional attributes
  - Address into street, city, zip
- Single-valued
  - Can have only a single value
  - Person has one social security number
- Multi-valued
  - Can have many values
  - Person may have several college degrees
- Derived
  - Can be derived with algorithm
  - Age can be derived from date of birth
- Attribute classification is driven by Client requirements
  - Phone Number?

Multivalued Attribute

- An attribute that has a list of values.
- For example:
  - Car colour may consist of body colour, trim colour, bumper colour.
- Crow's foot notation does not support multivalued attributes. Values are listed as a separate attribute.

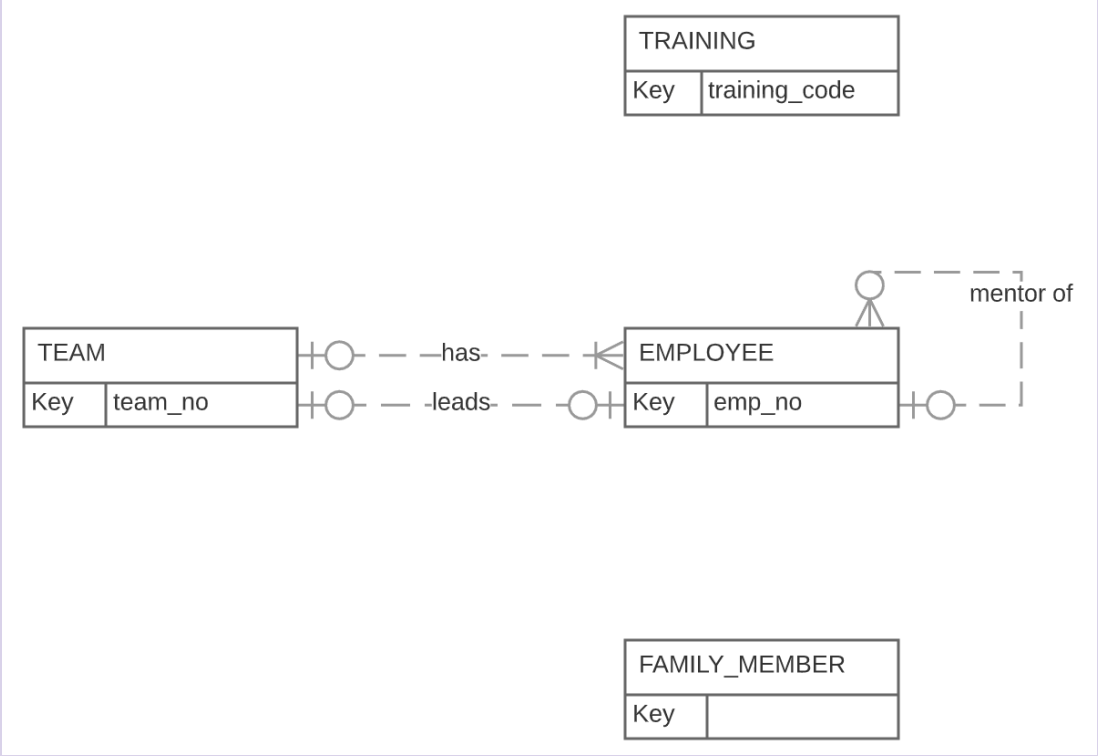


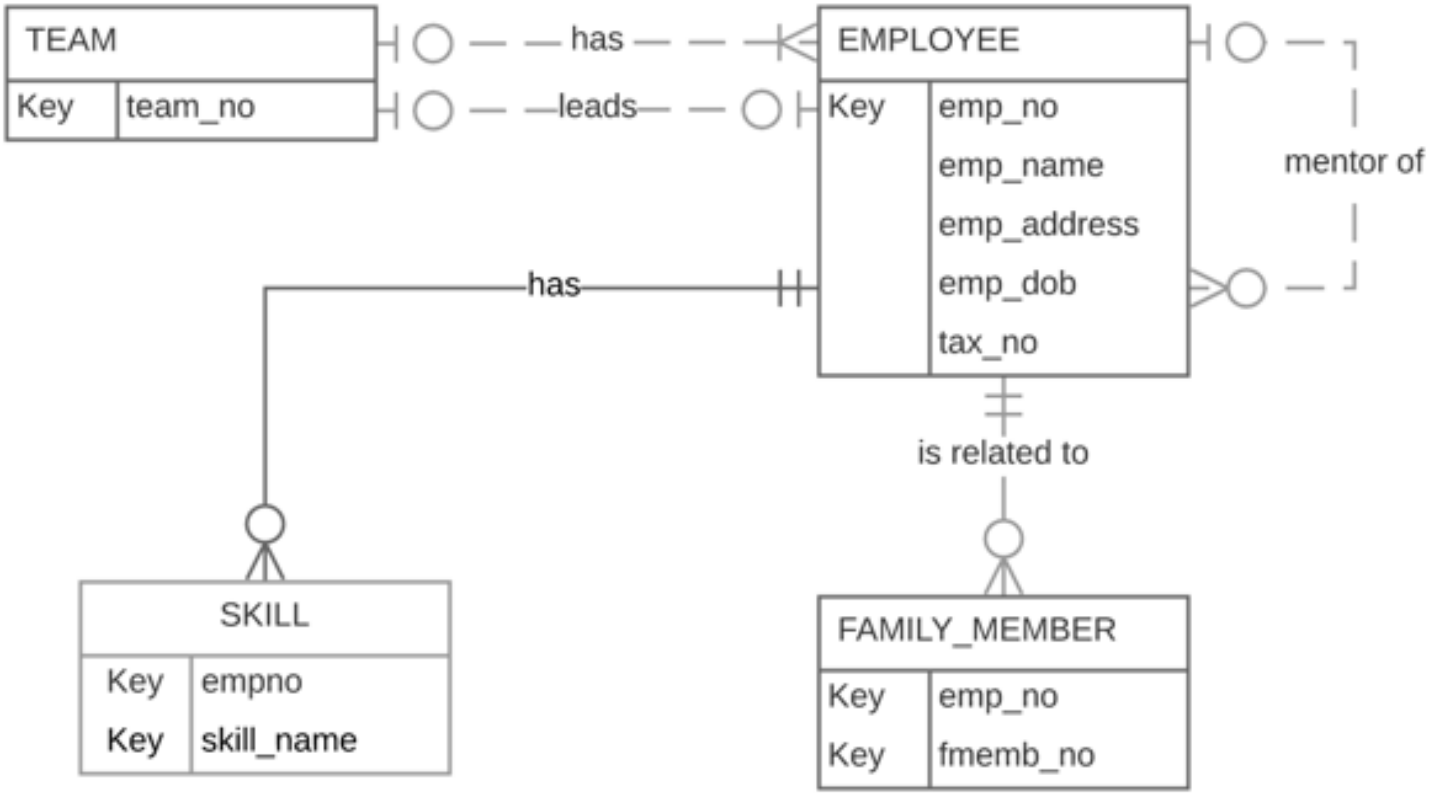
Resolving Multivalued Attributes



Student Activities

- Using the Monash Software Case Study, add attributes to your EMPLOYEE entity

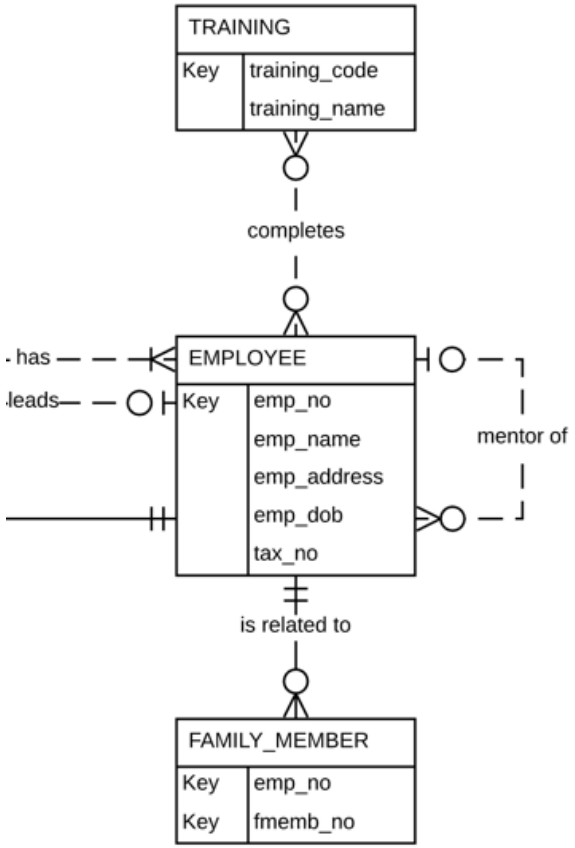



TEAM
Key
team_no
has
leads
EMPLOYEE
Key
emp_no
emp_name
emp_address
emp_dob
tax_no
mentor of
has
is related to
SKILL
Key
empno
Key
skill_name
FAMILY_MEMBER
Key
emp_no
Key
fmemb_no

..."the company provides several in-house training programs. The HR team needs to keep track of the details about who has done what. An employee can do several training programs.

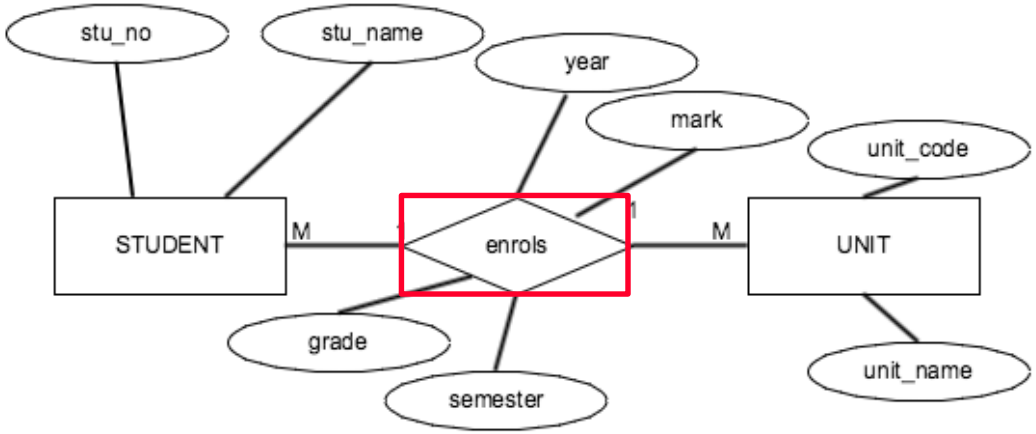
At the completion of a training, a certificate will be provided to the employee containing the training name and the completion date."...

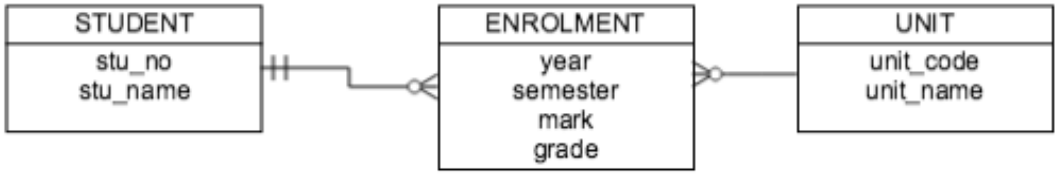
Incomplete model



Incomplete model

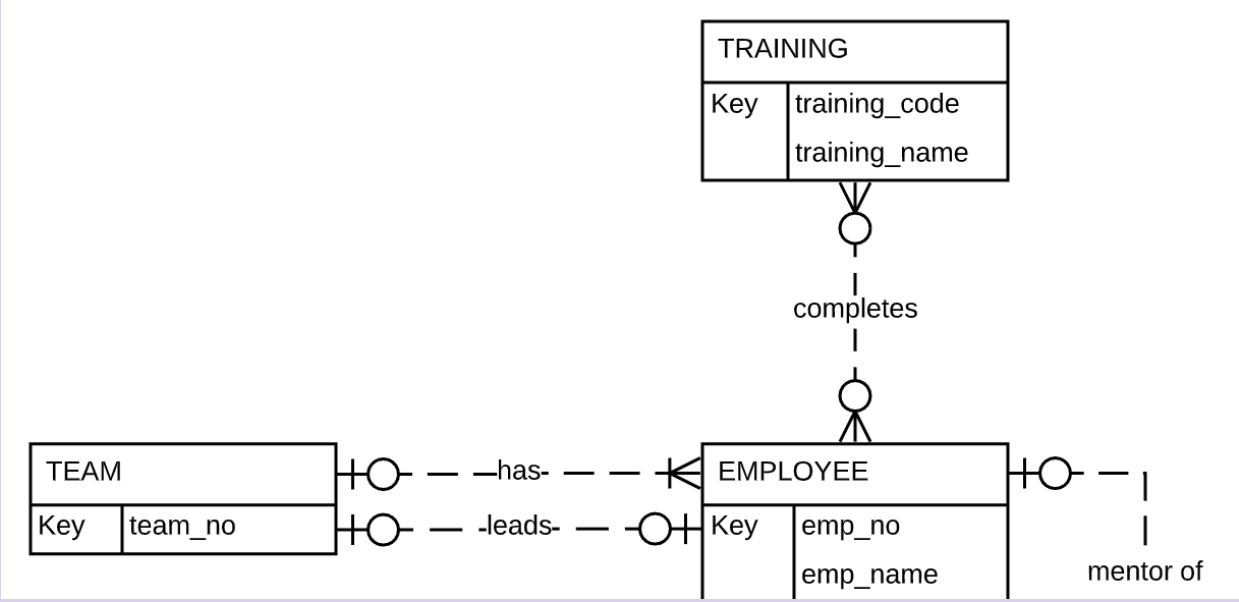
Associative (or Composite) Entity





Student Activities

- Using the Monash Software Case Study, add attribute to TRAINING EMPLOYEE

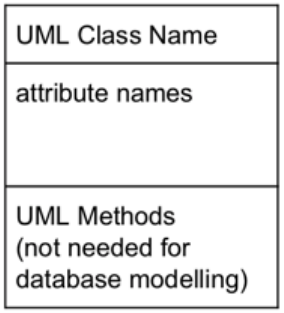


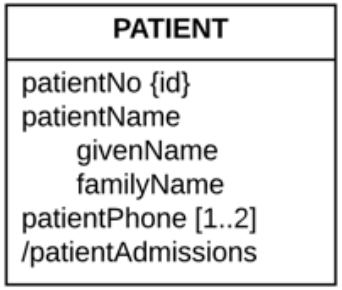
Unified Modeling Language (UML)

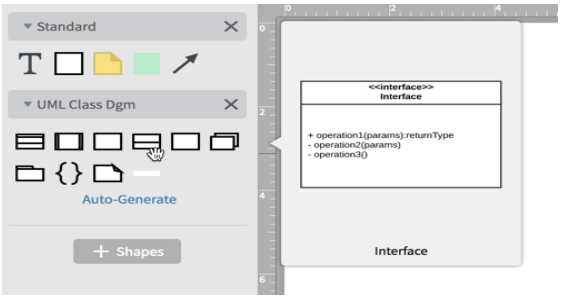
- The way that data is organised in a database is very different to the way it is organised in an OO program
  - eg. inheritance
- Use a **subset** of UML notation for database modelling
  - Conceptual
  - Logical
- Several vendors support Database Modelling via UML, some examples:
  - Star UML
  - Altova UModel
- Variety of standards adopted, not widely used in practice

UML Notation for the unit

Standard UML Diagram is used as the basic structure:







{id} - indicates KEY

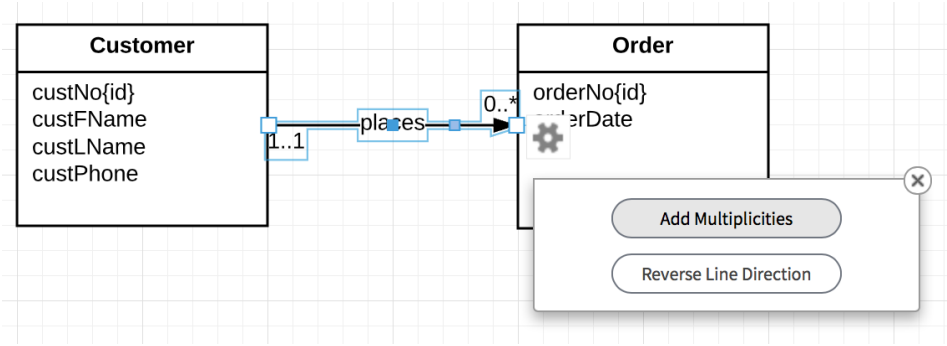
Indentation for composite attribute

[n..m] - multivalued attribute

/ - calculated attribute

use Interface shape as no methods being added

UML - Relationship



Relationship lines - directed line, arrowhead at M end,
add Multiplicities (minimal and maximal on each side)