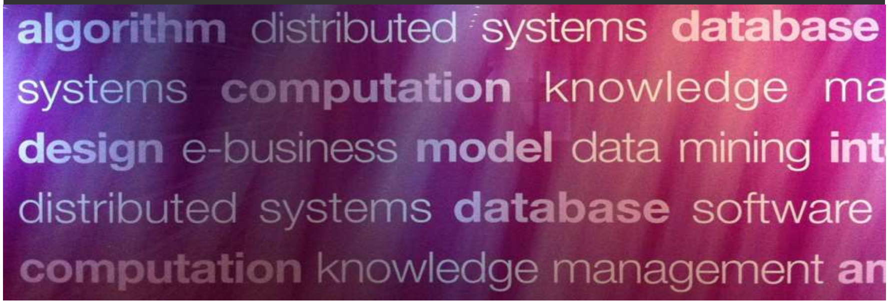
MONASH University

Information Technology

FIT3176 Advanced Database Design

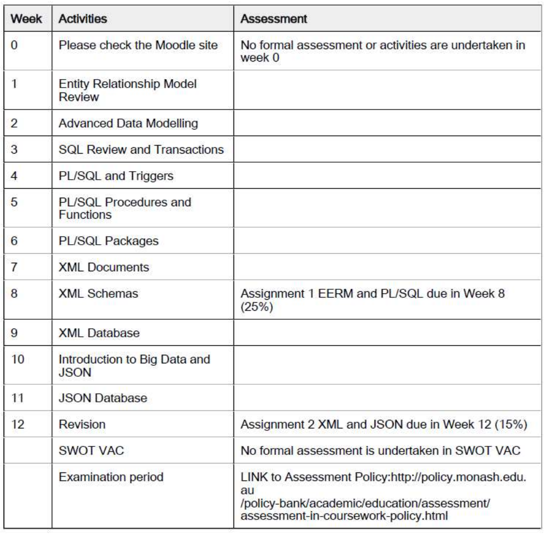
Topic 1: Review of Entity Relationship (ER) Modelling


algorithm distributed systems database
systems computation knowledge ma
design e-business model data mining int
distributed systems database software
computation knowledge management an

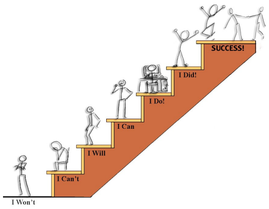
FIT3176 Staff Involved

- Lecturers
  - David Taniar (Chief Examiner)
    - Email: david.taniar@monash.edu
  - Minh Le
    - Email: minh.le@monash.edu
- Tutors
  - Minh Le
    - Studio: 3, 5
    - Email: minh.le@monash.edu
  - Huashun Li
    - Studio: 1, 2, 4
    - Email: huashun.li@monash.edu

Unit Structure



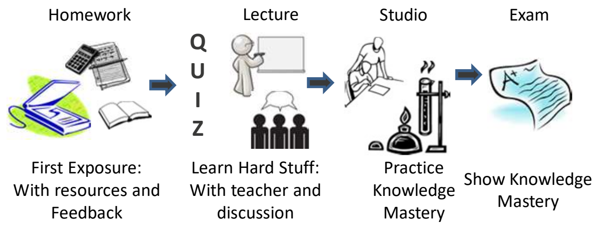
Week	Activities	Assessment
0	Please check the Moodle site	No formal assessment or activities are undertaken in week 0
1	Entity Relationship Model Review	
2	Advanced Data Modelling	
3	SQL Review and Transactions	
4	PL/SQL and Triggers	
5	PL/SQL Procedures and Functions	
6	PL/SQL Packages	
7	XML Documents	
8	XML Schemas	Assignment 1 EERM and PL/SQL due in Week 8 (25%)
9	XML Database	
10	Introduction to Big Data and JSON	
11	JSON Database	
12	Revision	Assignment 2 XML and JSON due in Week 12 (15%)
	SWOT VAC	No formal assessment is undertaken in SWOT VAC
	Examination period	LINK to Assessment Policy:http://policy.monash.edu.au /policy-bank/academic/education/assessment/ assessment-in-coursework-policy.html



Assessment Breakdown

- Please READ the FIT3176 study guide carefully
- Non-exam assessment (40%)
  - Assignment 1 Extended Entity Relationship Modelling Design and PL/SQL (25%)
    - Designing a database using advanced data modelling approach
    - Triggers, Procedures, Functions and Package
    - Due Friday of week 8 Individual Task
  - Assignment 2 XML and JSON (15%)
    - Due Friday of week 12 Individual Task
- Exam assessment (60%)
  - Exam will be 2 hours (closed book).
  - Exam will cover all learning outcomes stated in our unit guide.

Peer Instruction



Homework → QUIZ → Lecture → Studio → Exam

First Exposure: With resources and Feedback

Learn Hard Stuff: With teacher and discussion

Practice Knowledge Mastery

Show Knowledge Mastery

- Using MARS
  1. Visit http://mars.mu on your internet enabled device
  2. Log in using your Authcate details (not required if you're already logged in to Monash)
  3. Touch the + symbol
  4. Enter the code for FIT3176: YE10IP
  5. Answer questions when they pop up.

If the following equations are true:

5 + 3 = 28

9 + 1 = 810

8 + 6 = 214

5 + 4 = 19

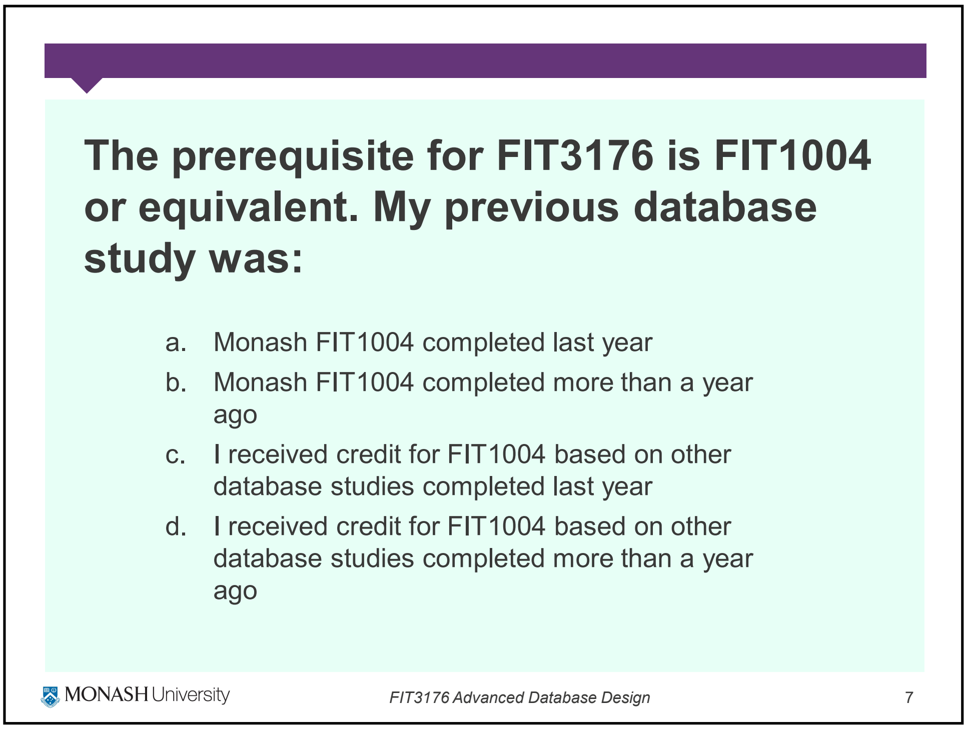
What is 3 + 2 ?

a. 5

b. 15

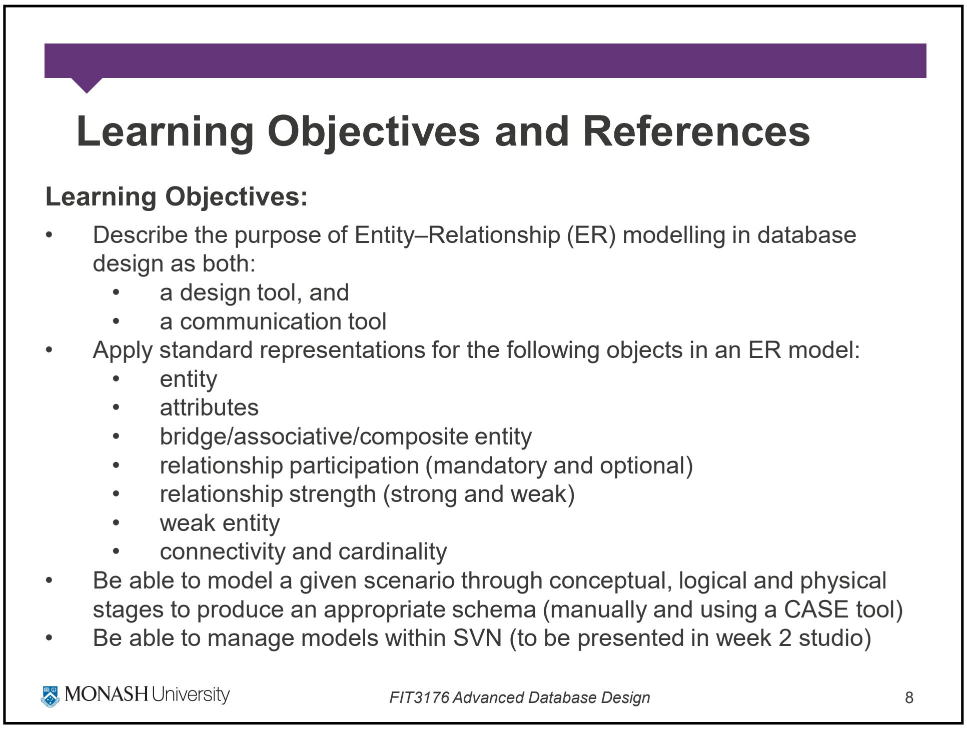
c. 11

d. 55



The prerequisite for FIT3176 is FIT1004 or equivalent. My previous database study was:

a. Monash FIT1004 completed last year
b. Monash FIT1004 completed more than a year ago
c. I received credit for FIT1004 based on other database studies completed last year
d. I received credit for FIT1004 based on other database studies completed more than a year ago



Learning Objectives and References

Learning Objectives:

- Describe the purpose of Entity–Relationship (ER) modelling in database design as both:
  - a design tool, and
  - a communication tool
- Apply standard representations for the following objects in an ER model:
  - entity
  - attributes
  - bridge/associative/composite entity
  - relationship participation (mandatory and optional)
  - relationship strength (strong and weak)
  - weak entity
  - connectivity and cardinality
- Be able to model a given scenario through conceptual, logical and physical stages to produce an appropriate schema (manually and using a CASE tool)
- Be able to manage models within SVN (to be presented in week 2 studio)

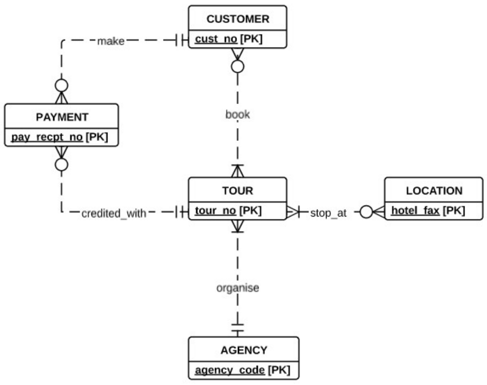
Learning Objectives and References

References:

- Coronel & Morris, Database Systems: Design, Implementation & Management, 12th Edition 2016, Thomson Course Technology. Chapter 4, Chapter 7 (7.1 – 7.2), Chapter 9 (9.6)
- Coronel, Morris, & Rob, Database Systems: Design, Implementation & Management, 11th Edition 2015, Thomson Course Technology. Chapter 4, Chapter 7 (7.1 – 7.2), Chapter 9 (9.6)

The slides for this week are based on the material presented in FIT1004 Data Management

Stage 1: Conceptual Model



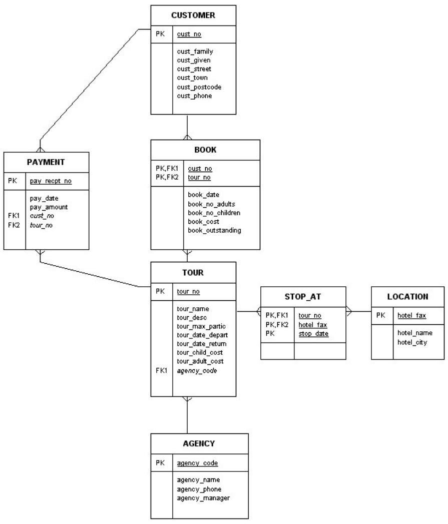
Expressed via **Entity Relationship Diagram (ERD)**

Sometimes non-key attributes are also shown

Representation:

generalised, non implementation model specific (eg. NO FK's) and non DBMS specific

Stage 2: Logical Model



Relational **Logical Model**

Representation:

now specific to a implementation model, here Relational (FK's have been added) but not specific to a particular DBMS (e.g., Oracle)

Expressed visually ERD like structures.

Stage 3: Physical Model

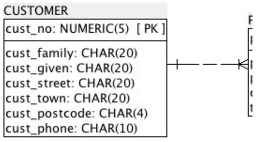
Expressed as a

DBMS Schema:

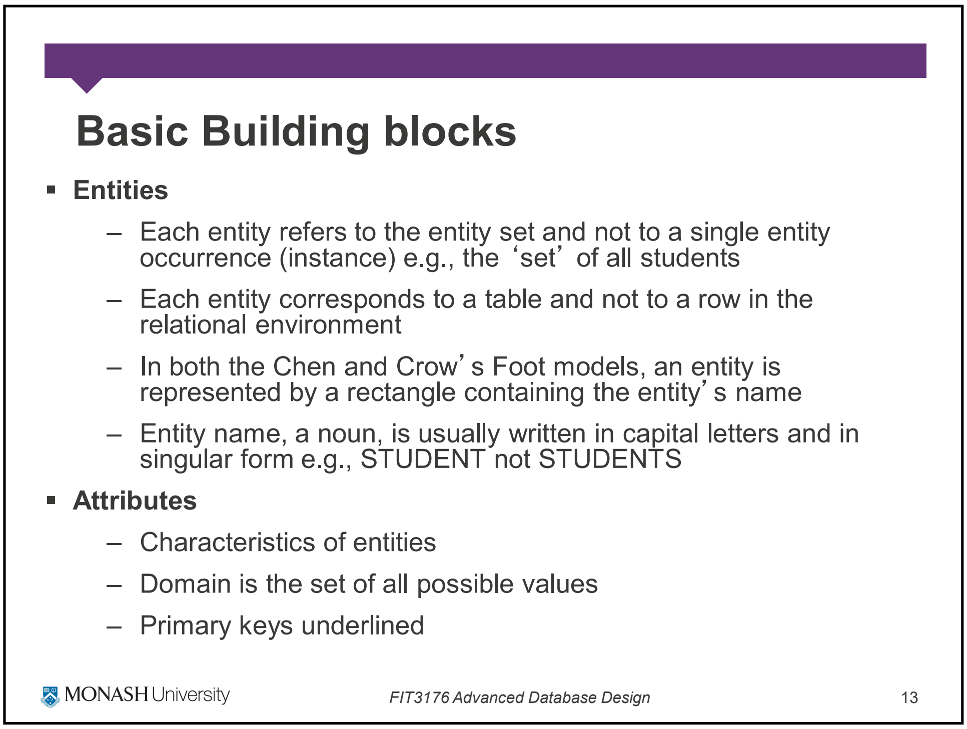
```
CREATE TABLE CUSTOMER (
   cust_no       NUMBER(5) NOT NULL,
   cust_family    CHAR(20) NOT NULL,
   cust_given     CHAR(20) NOT NULL,
   cust_street    CHAR(20) NOT NULL,
   cust_town      CHAR(20) NOT NULL,
   cust_postcode  CHAR(4) NOT NULL,
   cust_phone     CHAR(10),
CONSTRAINT pk_CUSTOMER
PRIMARY KEY (cust_no)); …etc
```

Or

Visually using ERD like structures:

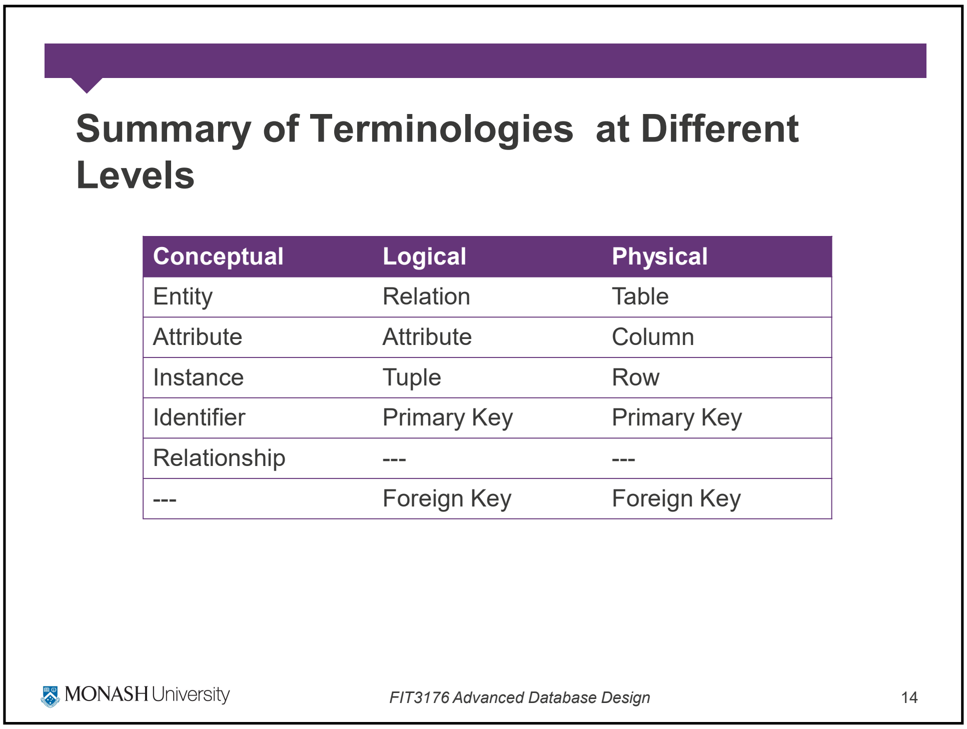


Representation now specific to a particular relational DBMS, here Oracle. Schema file can be run against database to create table structures



Basic Building blocks

- **Entities**
  - Each entity refers to the entity set and not to a single entity occurrence (instance) e.g., the ‘set’ of all students
  - Each entity corresponds to a table and not to a row in the relational environment
  - In both the Chen and Crow’s Foot models, an entity is represented by a rectangle containing the entity’s name
  - Entity name, a noun, is usually written in capital letters and in singular form e.g., STUDENT not STUDENTS
- **Attributes**
  - Characteristics of entities
  - Domain is the set of all possible values
  - Primary keys underlined

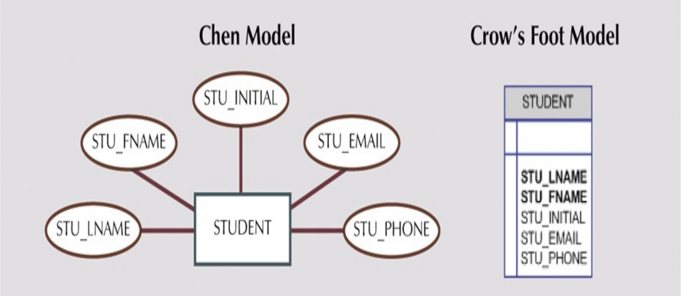


Summary of Terminologies at Different Levels

Conceptual	Logical	Physical
Entity	Relation	Table
Attribute	Attribute	Column
Instance	Tuple	Row
Identifier	Primary Key	Primary Key
Relationship	---	---
---	Foreign Key	Foreign Key

Attributes

FIGURE 4.1 The attributes of the STUDENT entity



Note no PK has been selected/shown as yet

Types of Attributes

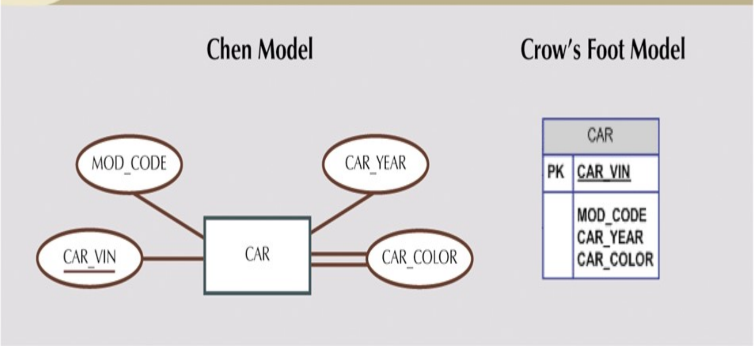
- Simple
  - Cannot be subdivided
  - Age, sex, marital status
- Composite
  - Can be subdivided into additional attributes
  - Address into street, city, zip
- Single-valued
  - Can have only a single value
  - Person has one social security number
- Multi-valued
  - Can have many values
  - Person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

Q1. The relational model requires that each cell in a relation is single-valued (atomic). Considering this requirement, what construct in an ER diagram cannot be implemented directly in the relational model (logical level)?

a. Composite key.
b. Composite attribute.
c. Multi-valued attribute.
d. Derived attribute.
e. More than one option is correct.

A Multivalued Attribute in an Entity

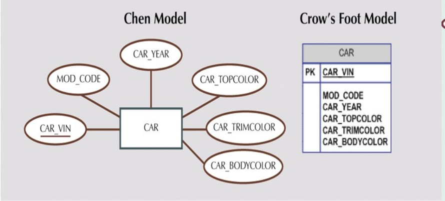
FIGURE 4.3 A multivalued attribute in an entity

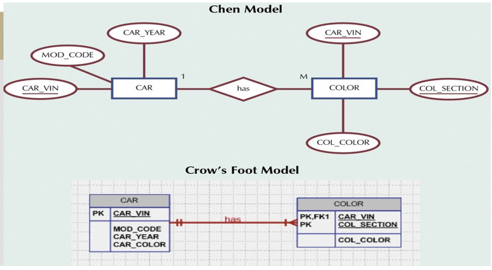


Resolving Multivalued Attribute Problems

- Although the conceptual model can depict multivalued attributes, you cannot implement them directly in the relational model. Instead we can:
  - Within original entity, create several new attributes, one for each component of the original multivalued attribute.
    - Can lead to major structural problems in the table
  - Or better, create a new entity composed of original multivalued attribute’s components

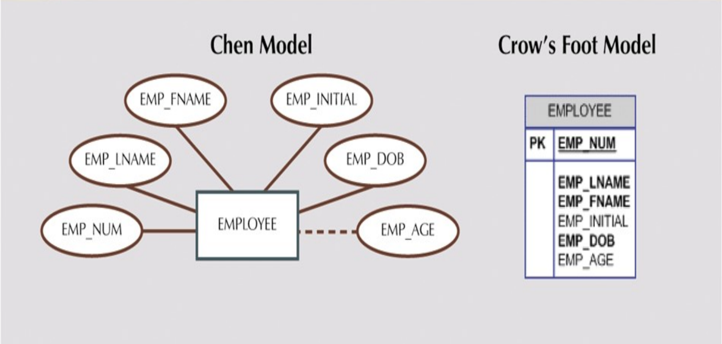
FIGURE 4.4 Splitting the multivalued attribute into new attributes

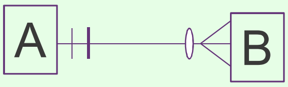




Derived Attribute

FIGURE 4.6 Depiction of a derived attribute





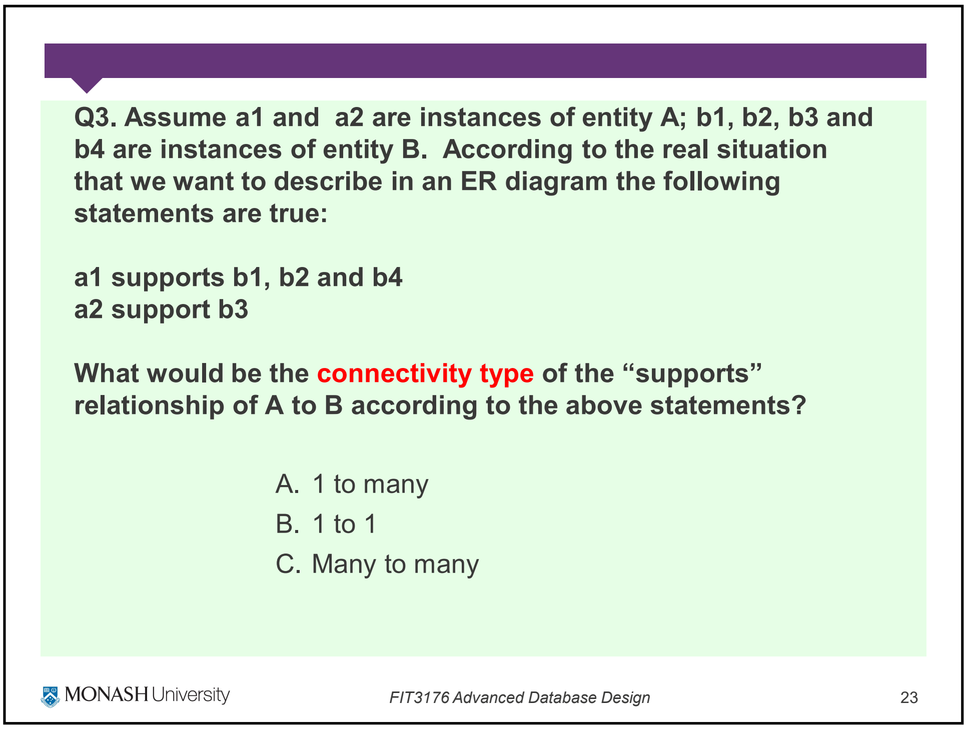
Q2. How many relationships are there on the above diagram?

What is the degree of the relationship (the number of entities participating in the relationship) ?

A. 1, unary
B. 2, binary
C. 1, binary
D. 3, ternary

Relationships

- Association between entities
- Connected entities are called participants
- Relationships between entities always operate in both directions
- Connectivity
  - describes relationship classification
    - 1:1
    - 1:M
    - M:N
- Cardinality
  - expresses number of entity occurrences associated with one occurrence of related entity
- Connectivities and Cardinalities are usually based on business rules.

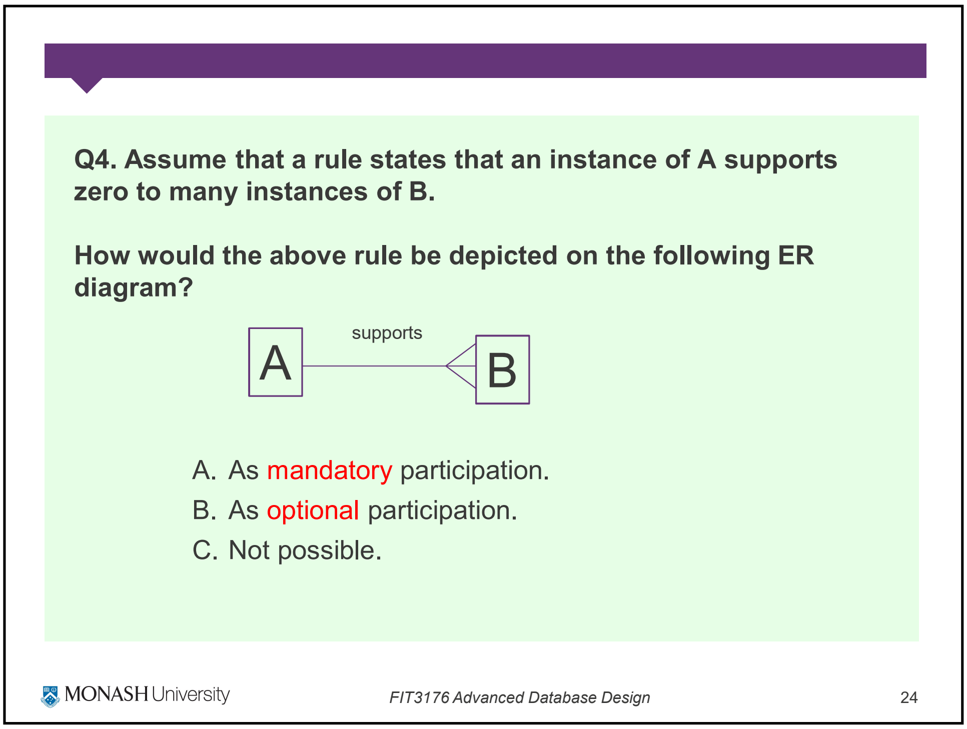


Q3. Assume a1 and a2 are instances of entity A; b1, b2, b3 and b4 are instances of entity B. According to the real situation that we want to describe in an ER diagram the following statements are true:

a1 supports b1, b2 and b4
a2 support b3

What would be the connectivity type of the "supports" relationship of A to B according to the above statements?

A. 1 to many
B. 1 to 1
C. Many to many



Q4. Assume that a rule states that an instance of A supports zero to many instances of B.

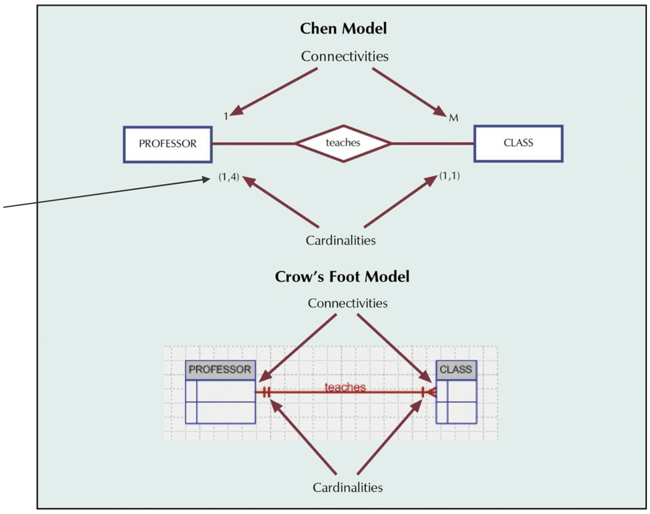
How would the above rule be depicted on the following ER diagram?

A
supports
B

A. As mandatory participation.
B. As optional participation.
C. Not possible.

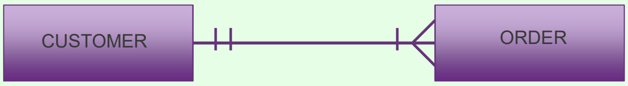
Relationship Connectivity and Cardinality

Uses (Min,Max) to express the specific number of entity occurrences associated with one occurrence of the related entity



Existence Dependence

- Existence dependence
  - An entity exists in a database only when it is associated with another related entity occurrence
    - e.g. CUSTOMER places ORDER
- Existence independence
  - Entity can exist apart from one or more related entities
  - Sometimes refers to such an entity as strong or regular entity.
- Many (most) entities on ER diagrams are existence dependant.



Q5. Where would you place the Foreign Key when you map the above ER diagram into the relational model?

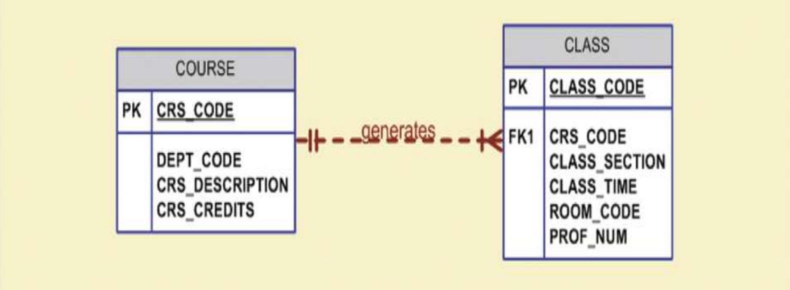
A. CUSTOMER
B. ORDER
C. Both CUSTOMER and ORDER.
D. None, no FK is needed.

Relationship Strength

- Weak (non-identifying) relationships
  - Exists if PK of related entity does not contain PK component of parent entity
  - **Use a broken line on the model**
- Strong (Identifying) Relationships
  - Exists when PK of related entity contains PK component of parent entity
  - **Use a solid line on the model**

A Weak (Non-Identifying) Relationship

FIGURE 4.8 A weak (non-identifying) relationship between COURSE and CLASS



A Strong (Identifying) Relationship

FIGURE 4.10 A strong (identifying) relationship between COURSE and CLASS

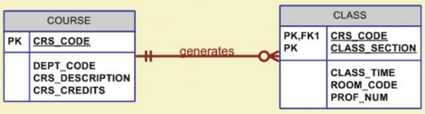
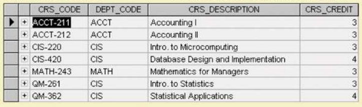
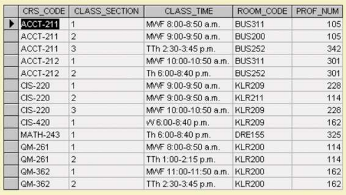


Table name: COURSE Database name: Ch04_TinyCollege_Alt



CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: CLASS



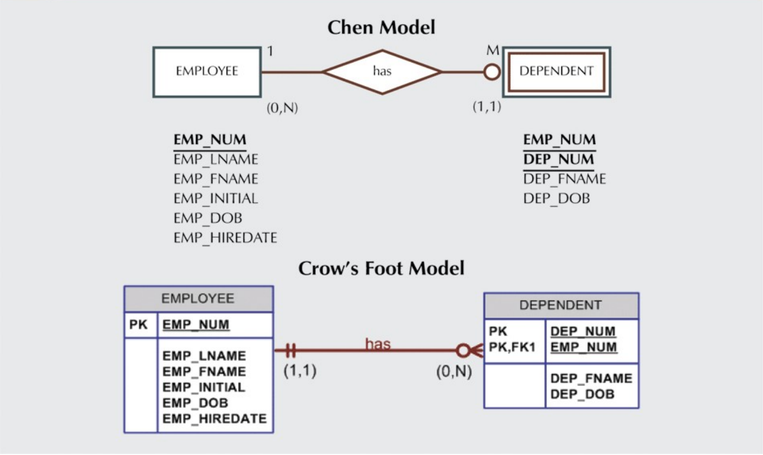
CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	PROF_NUM
ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325
QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162

Weak Entities

- A weak entity meets two conditions
  - Existence-dependent
    - Cannot exist without entity with which it has a relationship
  - Strong Relationship exists between entity and parent
    - **Child (weak entity) has primary key that is partially or totally derived from parent entity in relationship**
- Database designer usually **determines whether an entity can be described as weak based on business rules**
  - customer pays monthly account
    - PK payment_no, *or*
    - PK cust_no, date_paid

A Weak Entity in an ERD

FIGURE 4.11 A weak entity in an ERD

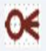
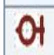


Relationship Participation

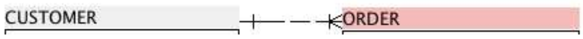
- Optional:
  - One entity occurrence *does not require* a corresponding entity occurrence in a particular relationship
- Mandatory:
  - One entity occurrence *requires* a corresponding entity occurrence in a particular relationship

Crow's Foot Participation

TABLE 4.3 Crow's Foot Symbols

CROW'S FOOT SYMBOL	CARDINALITY	COMMENT
	(0,N)	"Many" side is optional.
	(1,N)	"Many" side is mandatory.
	(1,1)	"1" side is mandatory.
	(0,1)	"1" side is optional.

In some variations of Crows Foot symbols (1,1) is shown as a single crossing line to represent *one and only one,* where possible show both lines



Crow's Foot Participation

FIGURE 4.14 CLASS is optional to COURSE

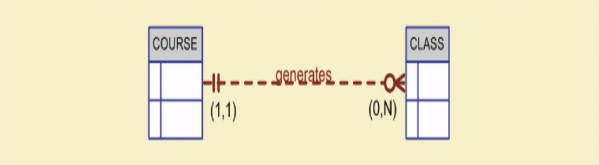
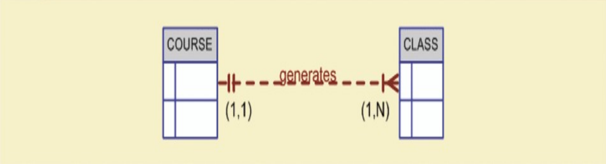


FIGURE 4.15 COURSE and CLASS in a mandatory relationship

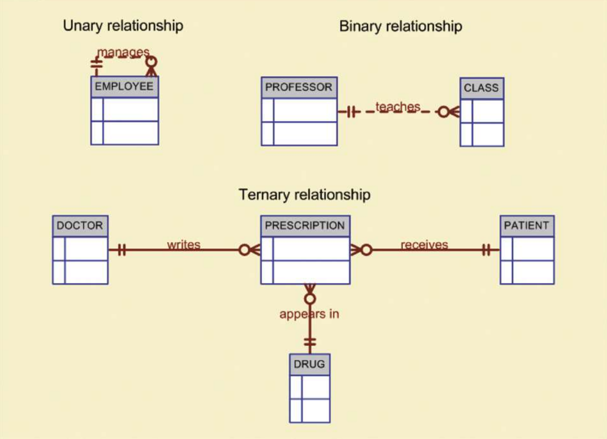


Relationship Degree

- Indicates number of associated entities or participants
- Unary relationship
  - Association is maintained within a single entity
- Binary relationship
  - Two entities are associated
- Ternary relationship
  - Three entities are associated

Three Types of Relationships

FIGURE 4.16 Three types of relationship degree



Recursive Relationships

- Relationship can exist between occurrences of the same entity set
- Naturally found within a unary relationship

FIGURE 4.18 An ER representation of recursive relationships

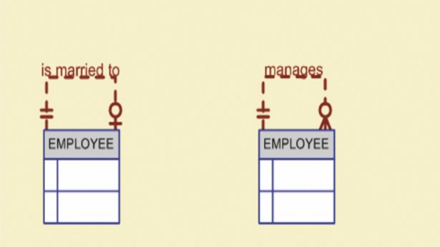
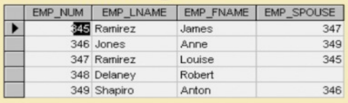


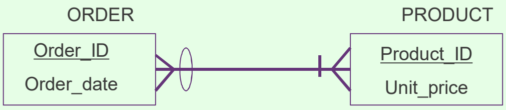
FIGURE 4.19 The 1:1 recursive relationship "EMPLOYEE is married to EMPLOYEE"

Database name: CH04_PartCo
Table name: EMPLOYEE_V1



EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_SPOUSE
345	Ramirez	James	347
346	Jones	Anne	349
347	Ramirez	Louise	345
348	Delaney	Robert	
349	Shapiro	Anton	346

Q6. What will be the Primary Key of the new created relation resulting from mapping the ER model below into a relational model?



A. The primary key of the ORDER table.
B. The primary key of the PRODUCT table.
C. The combination of primary keys of ORDER and PRODUCT.

Implementation of the M:N Recursive "PART Contains PART" Relationship

FIGURE 4.21 Implementation of the M:N recursive "PART contains PART" relationship

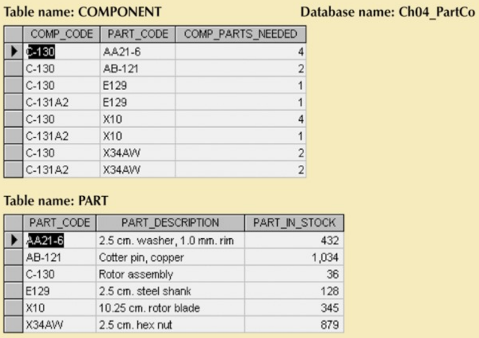


Table name: COMPONENT

Database name: Ch04_PartCo

COMP_CODE	PART_CODE	COMP_PARTS_NEEDED
C-130	AA21-6	4
C-130	AB-121	2
C-130	E129	1
C-131A2	E129	1
C-130	X10	4
C-131A2	X10	1
C-130	X34AW	2
C-131A2	X34AW	2

Table name: PART

PART_CODE	PART_DESCRIPTION	PART_IN_STOCK
AA21-6	2.5 cm. washer, 1.0 mm. rim	432
AB-121	Cotter pin, copper	1,034
C-130	Rotor assembly	36
E129	2.5 cm. steel shank	128
X10	10.25 cm. rotor blade	345
X34AW	2.5 cm. hex nut	879

COMPOSITE or ASSOCIATIVE Entity

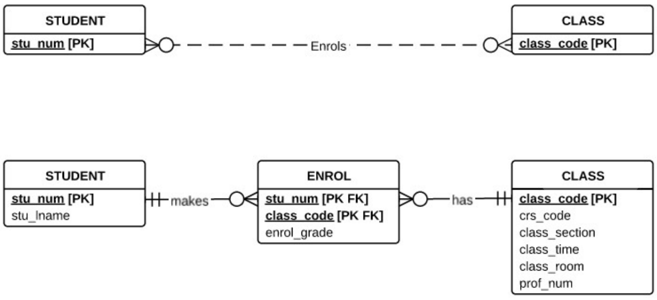
- Original CHEN ER model did not contain attributes on relationships
  - e.g. ORDERS --- are_for --- PRODUCT M:N need quantity ordered
    - where should this be placed
      - ORDERS, PRODUCT or ?
- Introduce bridging entity composed of keys of the two related entities, known as a COMPOSITE or ASSOCIATIVE or BRIDGING ENTITY
  - Note: associative entity is existence-dependent
- M:N relationship is equivalent to two 1:M relationships
  - An associative entity is identified as a strong (identifying) relationship, as indicated by the solid relationship lines, between parents and the associative entity.

The M:N Relationship Between STUDENT and CLASS

Conceptual Model (ERD)

Logical

Relational Model



STUDENT: stu_num [PK] — Enrols — CLASS: class_code [PK]

STUDENT: stu_num [PK], stu_lname — makes — ENROL: stu_num [PK FK], class_code [PK FK], enrol_grade — has — CLASS: class_code [PK], crs_code, class_section, class_time, class_room, prof_num

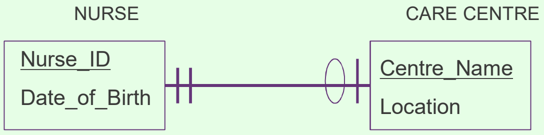
Software tools **modelling at the Logical level** do not allow the designer to draw M:N relations (two 1:M relationships must be used).

Conceptual ERD to Logical Model

BINARY RELATIONSHIPS:

- 1:1 relationship
  - Place primary key attributes/s of one entity into the other as foreign key (FK)
    - two entities on the logical model
  - Special case: *1:1 total* relationship (mandatory participation on both sides)
    - consolidate
      - one entity on the logical model
- 1:N relationship
  - place primary key attribute/s of one end into many end as foreign key (FK)
    - two entities on the logical model
- N:M relationship
  - create an intersection record, based on a composite entity
    - three entities on the logical model

Q7. Where would you place the Foreign Key when mapping the ER diagram below into the relational model?

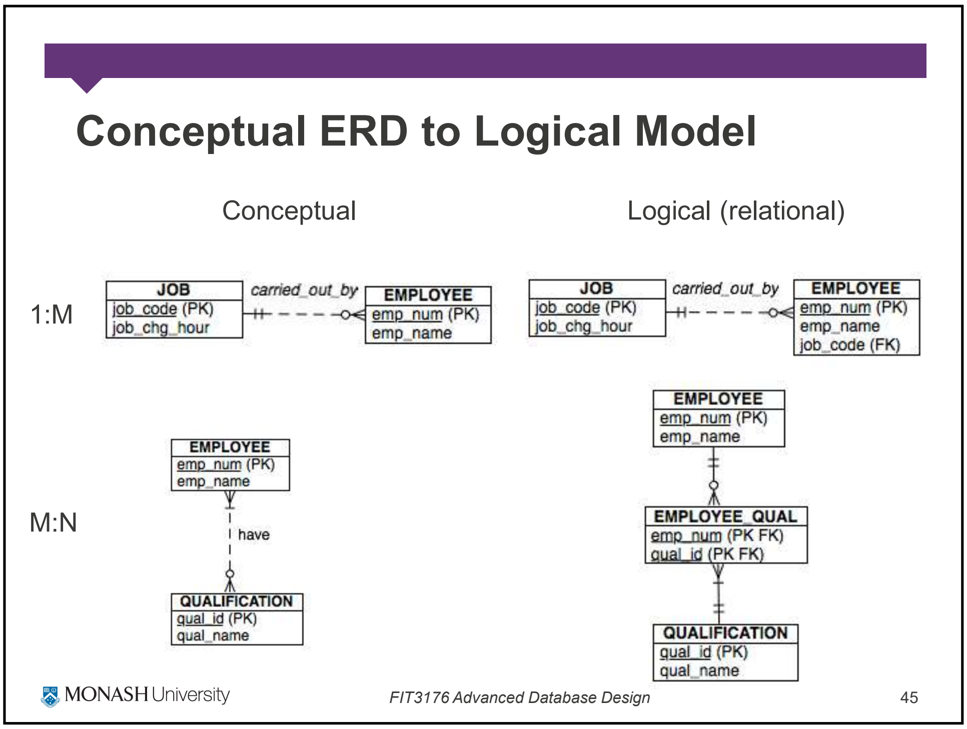


A. NURSE

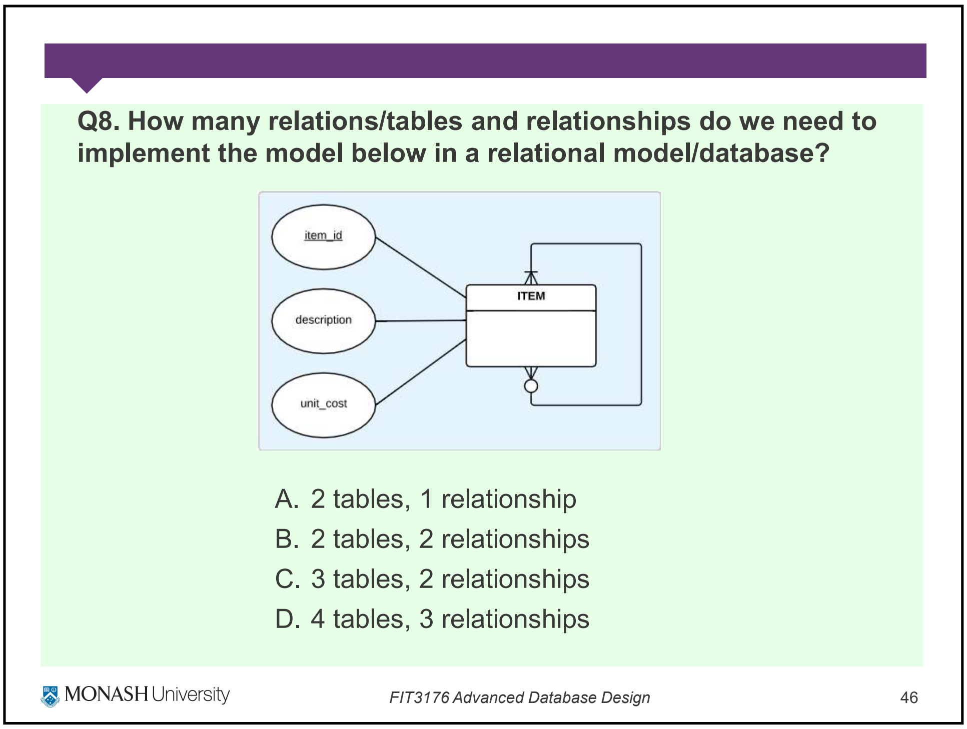
B. CARE CENTRE

C. Both NURSE and CARE CENTRE

D. No FK is needed.



Conceptual ERD to Logical Model



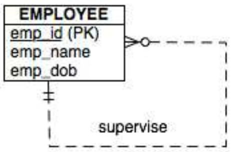
Q8. How many relations/tables and relationships do we need to implement the model below in a relational model/database?

A. 2 tables, 1 relationship
B. 2 tables, 2 relationships
C. 3 tables, 2 relationships
D. 4 tables, 3 relationships

Unary Relationships

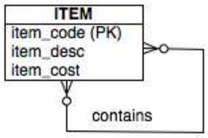
Unary 1:M

Conceptual

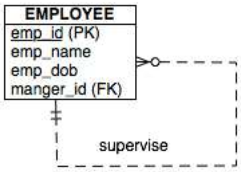


Unary M:N

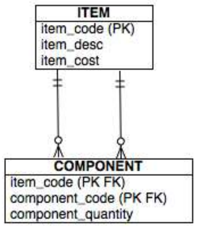
Conceptual



Logical

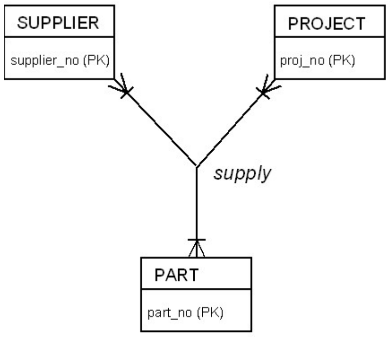


Logical

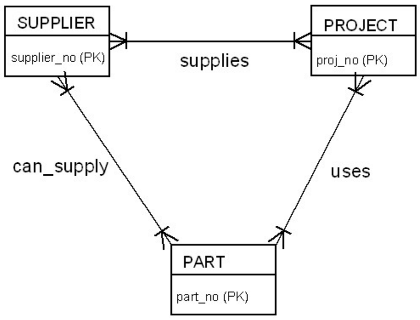


Ternary Relationships

Ternary



modelled as binary:

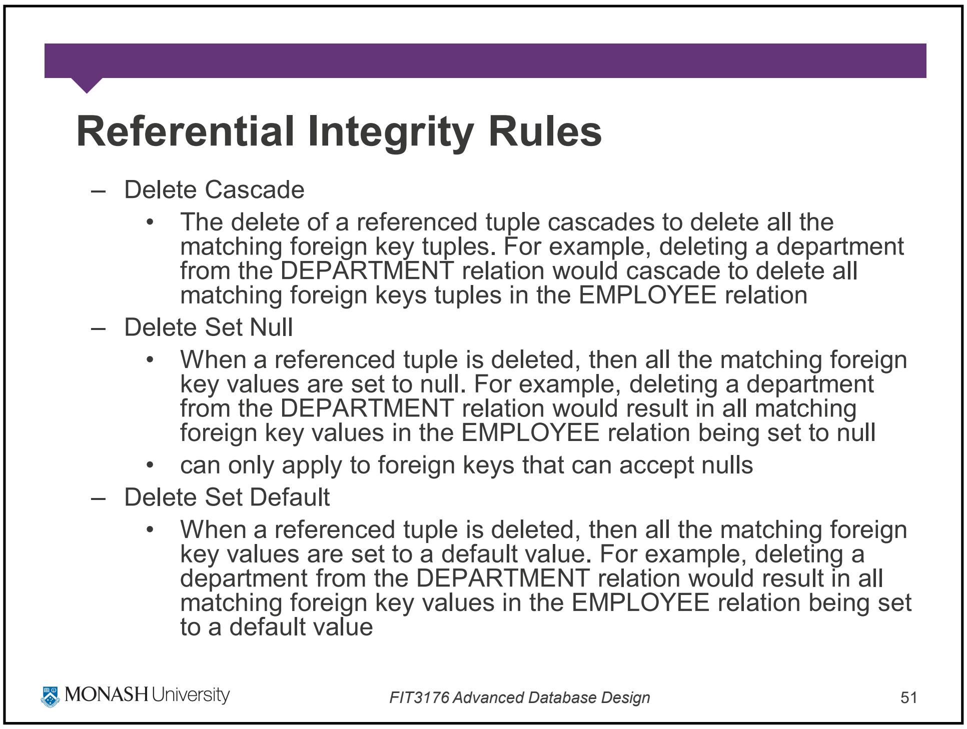


Ternary Relationships – model as binary relationships?

- Ternary represents more information than three binary relationships
- For example - Supplier 1 supplies Project 2 with Part 3 -
  - ternary
    - instance (supplier 1, project 2, part 3) exists
  - binaries
    - instances
      - (supplier1, project 2) (project 2, part 3) (supplier 1, part 3)
    - BUT does not imply (supplier 1, project 2, part 3)
- Model relationship via a **composite weak entity** relating all three entities together

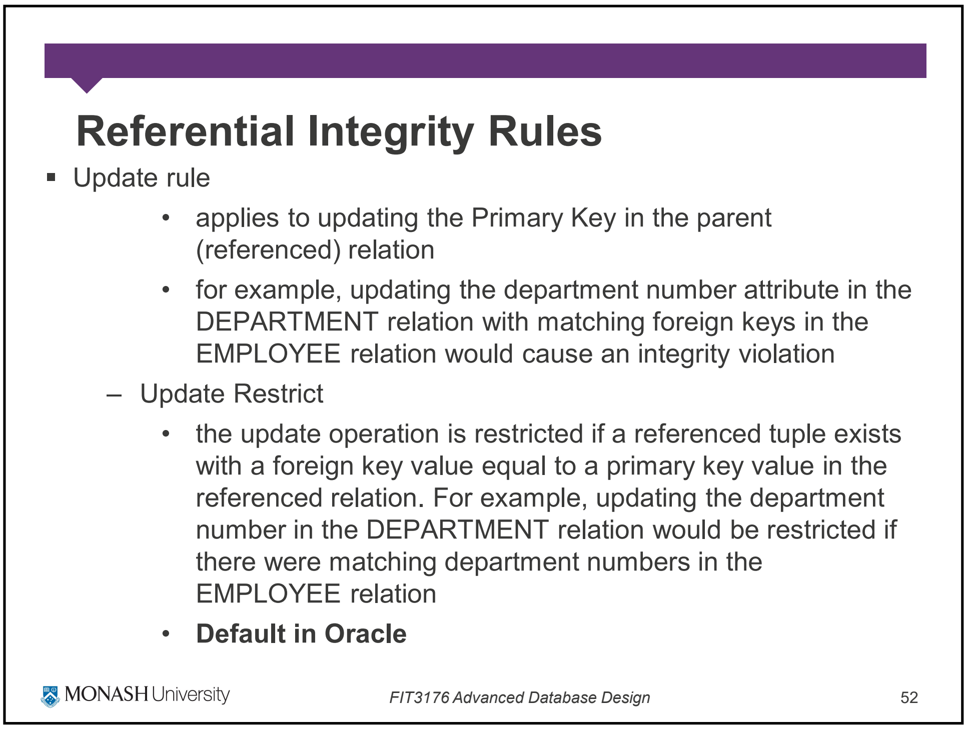
Referential Integrity Rules

- Delete rule
    - applies to deleting tuples from the parent (referenced) relation, that is the relation with the primary key
    - for example, deleting a department from the DEPARTMENT relation with matching foreign keys in the EMPLOYEE relation would cause **an integrity violation**
  - the delete can be restricted, cascaded, nullified or set to a default value
  - Delete Restrict
    - The delete operation is restricted or not allowed if a referenced tuple with foreign keys is the target of the delete. For example, deleting a department from the DEPARTMENT relation would be restricted if there were matching foreign keys in the EMPLOYEE relation.
    - **Default in Oracle**



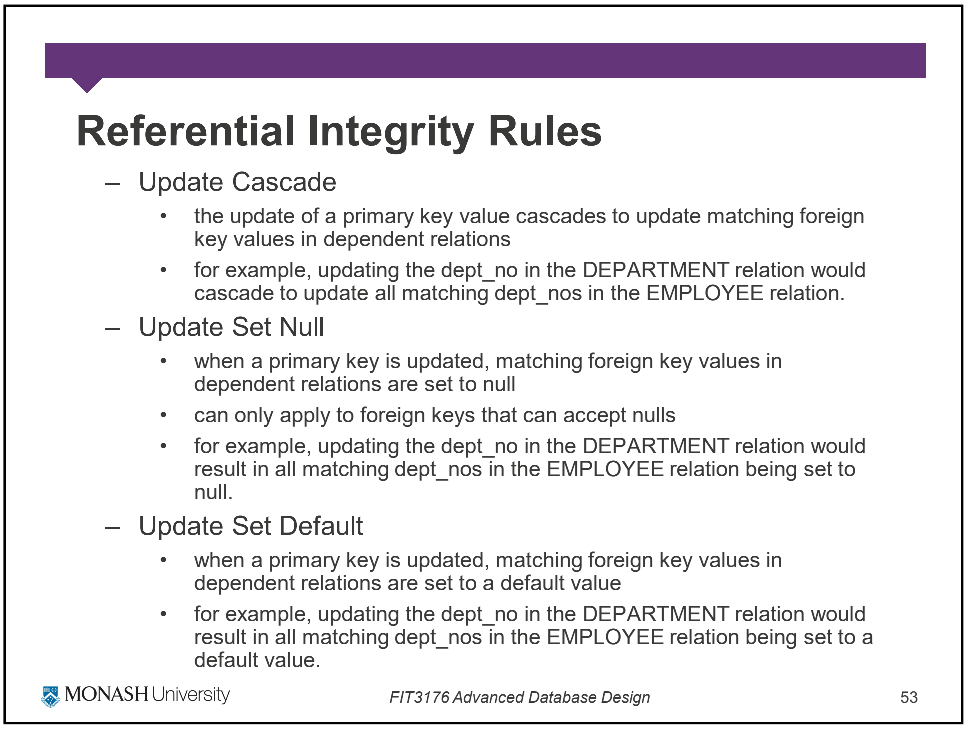
Referential Integrity Rules

- Delete Cascade
  - The delete of a referenced tuple cascades to delete all the matching foreign key tuples. For example, deleting a department from the DEPARTMENT relation would cascade to delete all matching foreign keys tuples in the EMPLOYEE relation
- Delete Set Null
  - When a referenced tuple is deleted, then all the matching foreign key values are set to null. For example, deleting a department from the DEPARTMENT relation would result in all matching foreign key values in the EMPLOYEE relation being set to null
  - can only apply to foreign keys that can accept nulls
- Delete Set Default
  - When a referenced tuple is deleted, then all the matching foreign key values are set to a default value. For example, deleting a department from the DEPARTMENT relation would result in all matching foreign key values in the EMPLOYEE relation being set to a default value



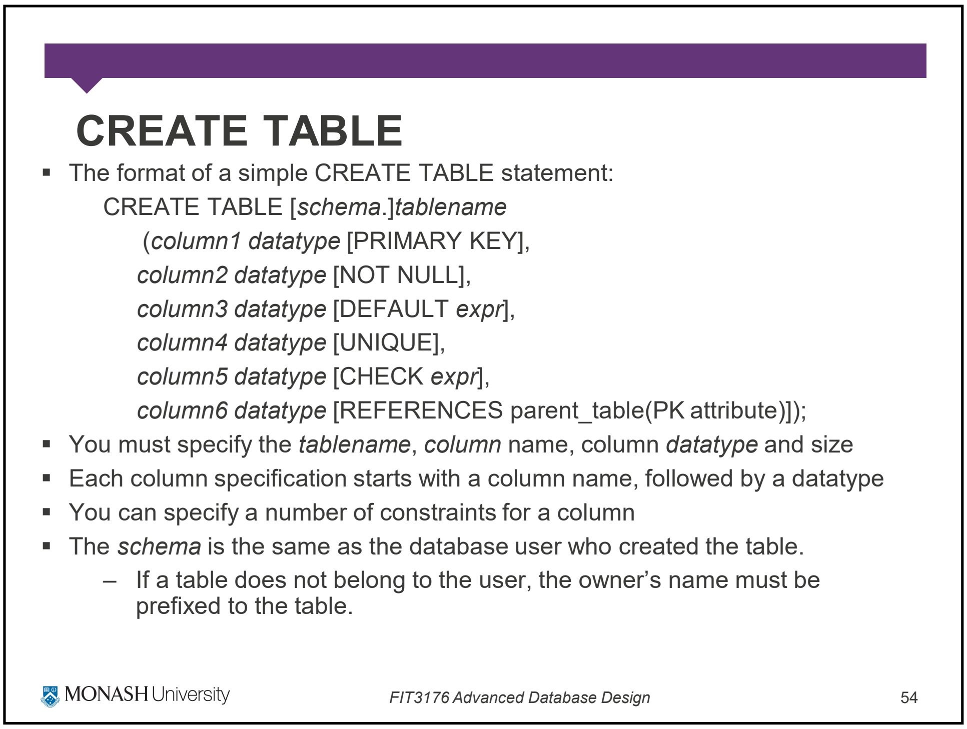
Referential Integrity Rules

- Update rule
  - applies to updating the Primary Key in the parent (referenced) relation
  - for example, updating the department number attribute in the DEPARTMENT relation with matching foreign keys in the EMPLOYEE relation would cause an integrity violation
  - Update Restrict
    - the update operation is restricted if a referenced tuple exists with a foreign key value equal to a primary key value in the referenced relation. For example, updating the department number in the DEPARTMENT relation would be restricted if there were matching department numbers in the EMPLOYEE relation
    - **Default in Oracle**



Referential Integrity Rules

- Update Cascade
  - the update of a primary key value cascades to update matching foreign key values in dependent relations
  - for example, updating the dept_no in the DEPARTMENT relation would cascade to update all matching dept_nos in the EMPLOYEE relation.
- Update Set Null
  - when a primary key is updated, matching foreign key values in dependent relations are set to null
  - can only apply to foreign keys that can accept nulls
  - for example, updating the dept_no in the DEPARTMENT relation would result in all matching dept_nos in the EMPLOYEE relation being set to null.
- Update Set Default
  - when a primary key is updated, matching foreign key values in dependent relations are set to a default value
  - for example, updating the dept_no in the DEPARTMENT relation would result in all matching dept_nos in the EMPLOYEE relation being set to a default value.



CREATE TABLE

- The format of a simple CREATE TABLE statement:

```
CREATE TABLE [schema.]tablename
  (column1 datatype [PRIMARY KEY],
  column2 datatype [NOT NULL],
  column3 datatype [DEFAULT expr],
  column4 datatype [UNIQUE],
  column5 datatype [CHECK expr],
  column6 datatype [REFERENCES parent_table(PK attribute)]);
```

- You must specify the *tablename*, *column* name, column *datatype* and size
- Each column specification starts with a column name, followed by a datatype
- You can specify a number of constraints for a column
- The *schema* is the same as the database user who created the table.
  - If a table does not belong to the user, the owner's name must be prefixed to the table.

CREATE TABLE Example

```
CREATE TABLE department
    (deptno  NUMBER(2)      CONSTRAINT d_PK  PRIMARY KEY,
     dname  VARCHAR2(10) CONSTRAINT d_name_NN NOT NULL
                                      CONSTRAINT d_dname_UN UNIQUE
                                      CONSTRAINT d_dname_CHK
                                                    CHECK (dname = upper(dname)),
     location VARCHAR2(8)    CONSTRAINT d_loc_NN NOT NULL
                                      CONSTRAINT d_loc_CHK
                                                    CHECK (location = upper(location)),
     mgr        NUMBER(4)      CONSTRAINT d_mgr_FK   REFERENCES employee) ;
```

- Oracle Naming Conventions
  - Must begin with a letter
  - Can be 1–30 characters long
  - Must contain only A–Z, a–z, 0–9, _, $, and #
  - Must not duplicate the name of another object owned by the same user
  - Must not be an Oracle Server reserved word

Oracle Data Types

CHAR(*size*)	**Fixed length character string (min 1, max 2000)**
VARCHAR2(*size*)	**Variable length character string (min 1, max 4000)**
VARCHAR(size)	*Currently synonymous with varchar2 datatype*
NUMBER(*p,s*)	**Number having precision *p* and scale *s*. The precision *p* can range from 1 to 38. The scale *s* can range from -84 to 127.**
DATE	**Date and time values. Stores century, year, month, date, hour, minute, and second**
LONG	**Variable-length character strings containing up to 2 gigabytes**
CLOB	**A character large object containing single-byte or multibyte characters up to 4 gigabytes**
RAW and LONG RAW	**Raw binary data**
BLOB	**A binary large object up to 4 gigabytes**
BFILE	**Contains a locator to a large binary file stored outside the database. Enables byte stream I/O access to external LOBs residing on the database server. Maximum size is 4 gigabytes.**

Table Constraints

- SQL CONSTRAINTS:
  - PRIMARY KEY
    - enforces both entity integrity and uniqueness
  - NOT NULL
    - ensures the column cannot contain NULL values
  - UNIQUE
    - ensures all values in the column are unique
  - CHECK
    - validates the values in the column to ensure they meet a specified condition
  - REFERENCES
    - ensures that non-NULL foreign key values match primary key values in the referenced (parent) table

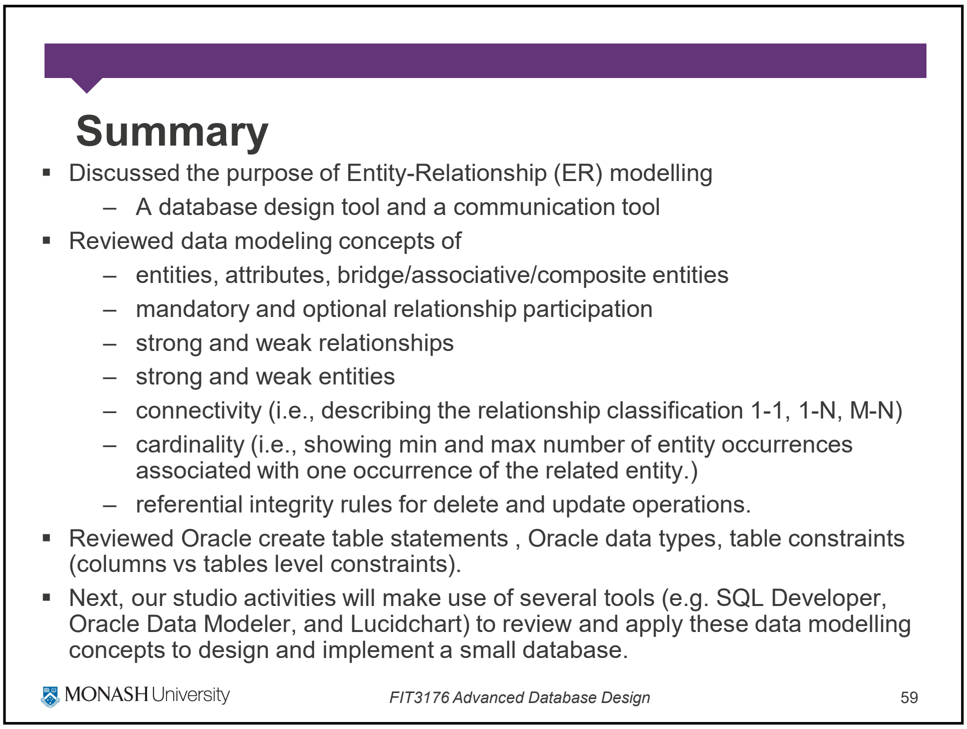
Table Constraints

- Constraints can be defined at one of two levels:
  - Column level

    column [CONSTRAINT constraint_name] constraint_type,
    - **references a single column**, and is defined within a specification for the owning column
    - can define any type of constraint
    - E.g. CUS_BALANCE NUMBER(9,2) DEFAULT 0.0
  - Table level

    column,

    [CONSTRAINT constraint_name] constraint_type (column, ...),
    - **references one or more columns**, and is defined separately from the definitions of the columns in the table
    - can define any type of constraint except NOT NULL
    - must be used for the definition of PRIMARY and FOREIGN KEYS that contain more than one attribute
    - E.g., CONSTRAINT CUS_UNQ UNIQUE (CUS_LNAME, CUS_FNAME)



Summary

- Discussed the purpose of Entity-Relationship (ER) modelling
  - A database design tool and a communication tool
- Reviewed data modeling concepts of
  - entities, attributes, bridge/associative/composite entities
  - mandatory and optional relationship participation
  - strong and weak relationships
  - strong and weak entities
  - connectivity (i.e., describing the relationship classification 1-1, 1-N, M-N)
  - cardinality (i.e., showing min and max number of entity occurrences associated with one occurrence of the related entity.)
  - referential integrity rules for delete and update operations.
- Reviewed Oracle create table statements , Oracle data types, table constraints (columns vs tables level constraints).
- Next, our studio activities will make use of several tools (e.g. SQL Developer, Oracle Data Modeler, and Lucidchart) to review and apply these data modelling concepts to design and implement a small database.