MONASH University

MONASH
INFORMATION
TECHNOLOGY

Database Design II: Logical Modelling

Reference

Several of the examples and diagrams used this session have been taken from:

Hoffer, J. A. , Prescott, M. B. & McFadden, F. R. “Modern Database Management”

Step 2 (and 3) of the Design Process

- Step 1 Conceptual Model (session 2)
  - Database Model independent
- Step 2 Logical Model (this session)
  - Select which type (model) of database you wish to implement your conceptual model in
    - Network, Relational, OO, XML, NoSQL, ...
  - Database model dependent
- Step 3 Physical Model
  - Select which specific vendor for your chosen model you will implement in
    - Oracle, MySQL, IBM DB2, SQL Server, ...
  - Database vendor dependent
  - Final output schema file to implement model (for relational model a set of tables)

Summary of Terminologies at Different Levels

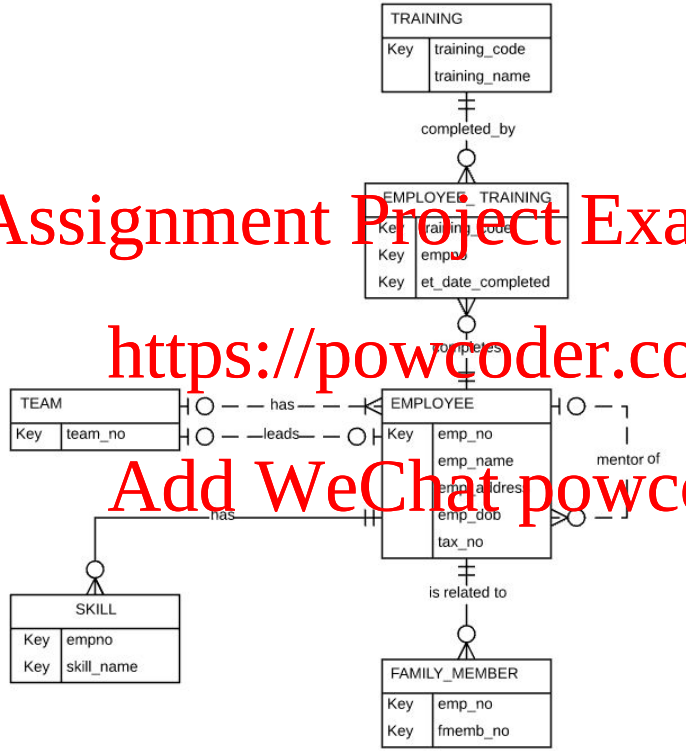
Conceptual	Logical (Relational)	Physical (Relational)
Entity	Relation	Table
Attribute	Attribute	Column
Instance	Tuple	Row
Identifier	Primary Key	Primary Key
Relationship	---	---
---	Foreign Key	Foreign Key

Recap Session 3 Relational Model Characteristics

- Each relation must have a unique name
- Each attribute of a relation must have a distinct name within the relation
- An attribute cannot be multivalued (consist of repeating values)
- All values of an attribute need to be from the same domain
- The order of attributes and tuples in a relation is immaterial
- Each relation must have a primary key
- Logical (not physical) connections are made between relations by virtue of primary/foreign keys

Revisit - Session 2 Conceptual Model

All required attributes shown



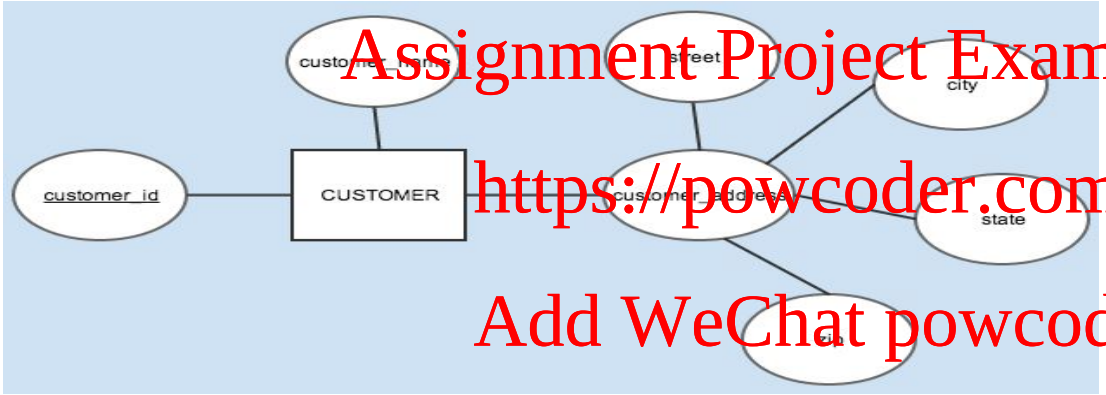
Transforming ER diagrams into relations (mapping conceptual level to logical level)

- Essentially
  - KEY to PK
  - Represent relationships with PK/FK pairs
- The steps are:
  - Map strong (regular) entities
  - Map weak entities
  - Map binary relationships
  - Map associative entities
  - Map unary relationships
  - Map ternary relationships
  - *Map supertype/subtype relationships (is not part of this unit).*

Map Regular Entities

- Composite Attributes
  - When the regular entity type contains a composite attribute, only the simple component attributes of the composite attribute are included in the new relation.
  - Compared to composite attributes, simple attributes not only improve data accessibility but also help in maintaining data quality
  - Client input needed in some cases to determine if to be left as simple or broken into components

Mapping a Composite Attribute



CUSTOMER	
P *	cust_id
*	cust_name
*	cust_street
*	cust_city
*	cust_state
*	cust_zip

Monash Software Case Study

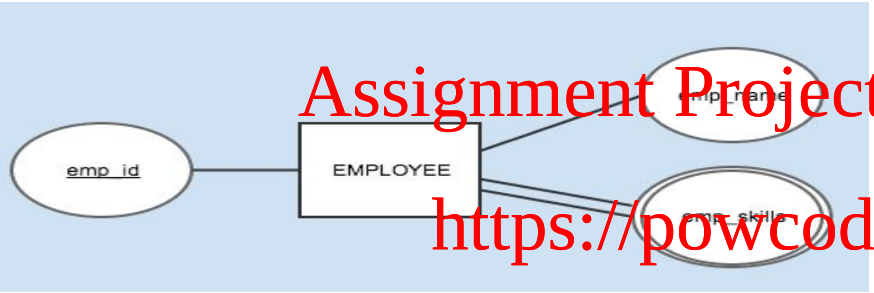
** = not null (must have value)*

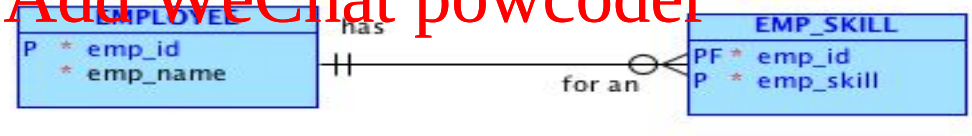
EMPLOYEE	
P *	emp_no
	emp_fname
	emp_lname
*	emp_street
*	emp_town
*	emp_pcode
*	emp_dob
	emp_taxno

Map Regular Entities

- Multivalued Attribute
  - When the regular entity type contains a multivalued attribute, two new relations are created.
  - The first relation contains all the attributes of the entity type except the multivalued attribute itself.
  - The second relation contains two attributes that form the PK. One of the attributes is the PK from the first relation, which becomes the FK in the second relation and the other is the multivalued attribute.
  - There can also be non key attributes in the second relation depending upon the data requirements.

Mapping a Multi valued Attribute

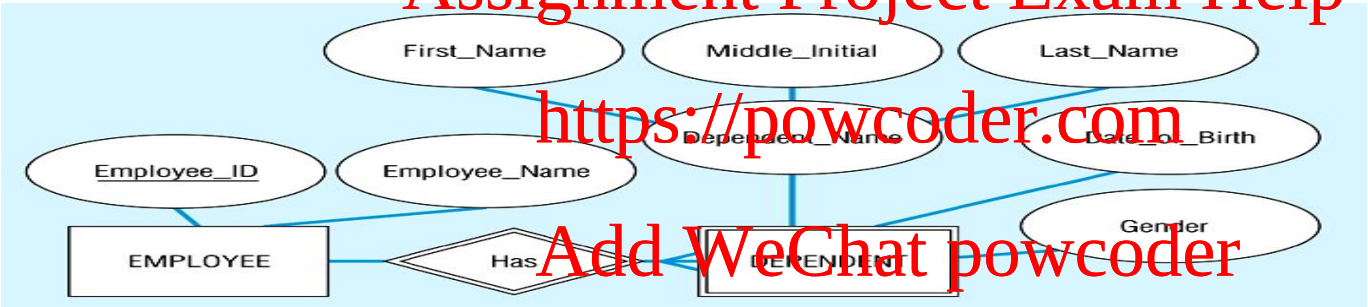


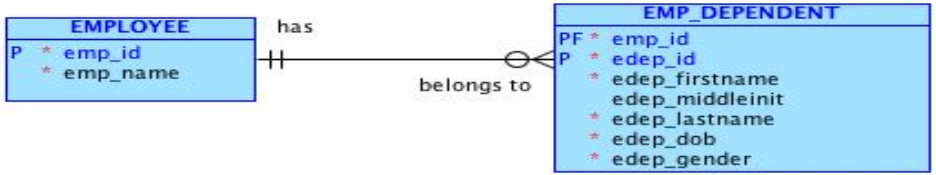


Is there a better solution than the one shown above?
What are the issues here - this was partially discussed in session 2

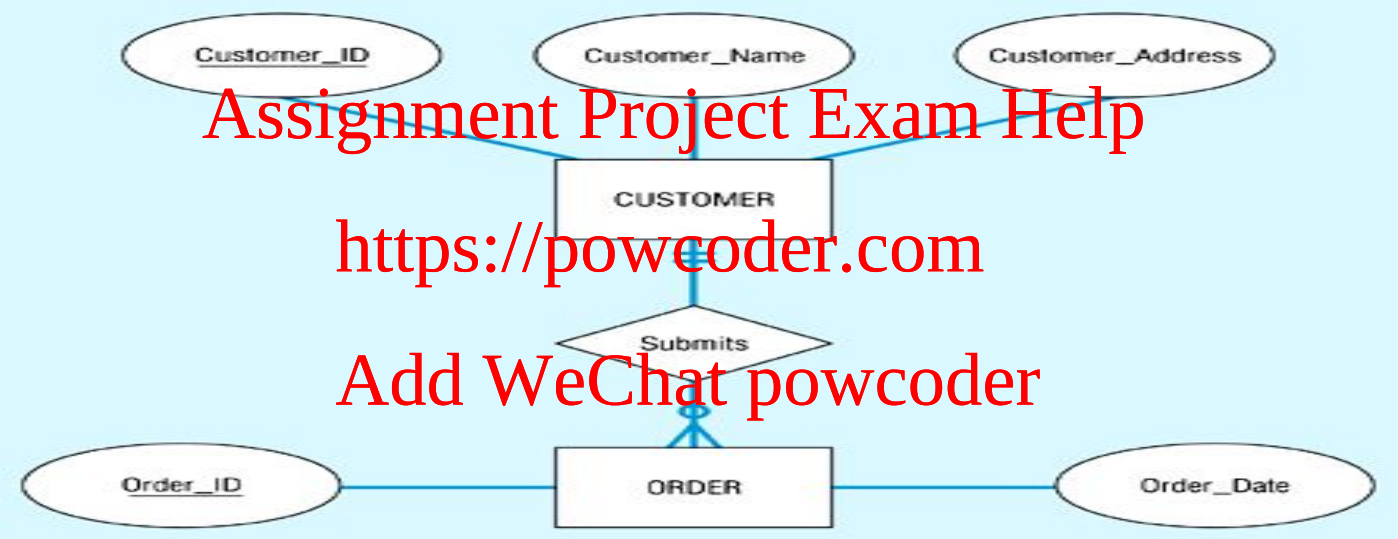
Mapping a Weak Entity

- For each weak entity type, create a new relation and include all of the simple attributes as attributes of this relation. The PK of the identifying relation is also included as the FK in this new relation.





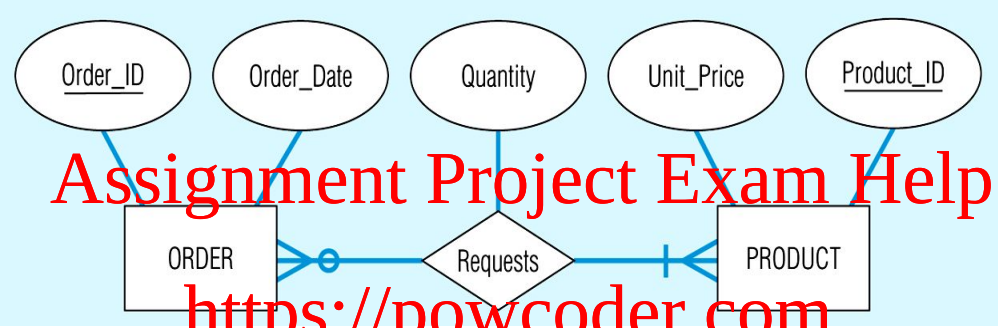
Mapping a 1:M Binary Relationship

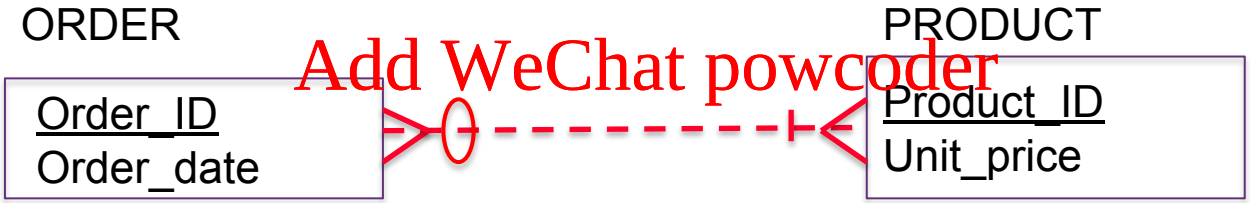


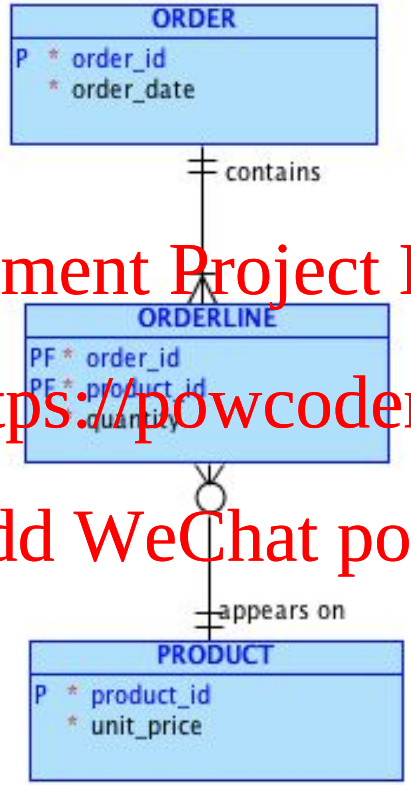
Map Binary Relationships (1:M)

For each 1:M binary relationship, first create a relation for each of the two entity types participating in the relationship. Then include the PK attribute (or attributes) of the entity on the one-side of the relationship as the FK on the many-side of the relationship.

Mapping a M:N Binary Relationship



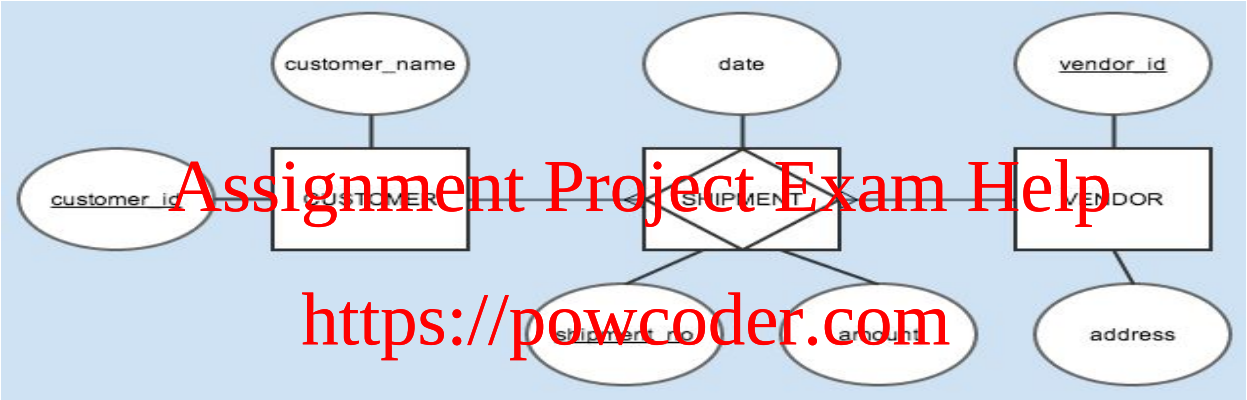




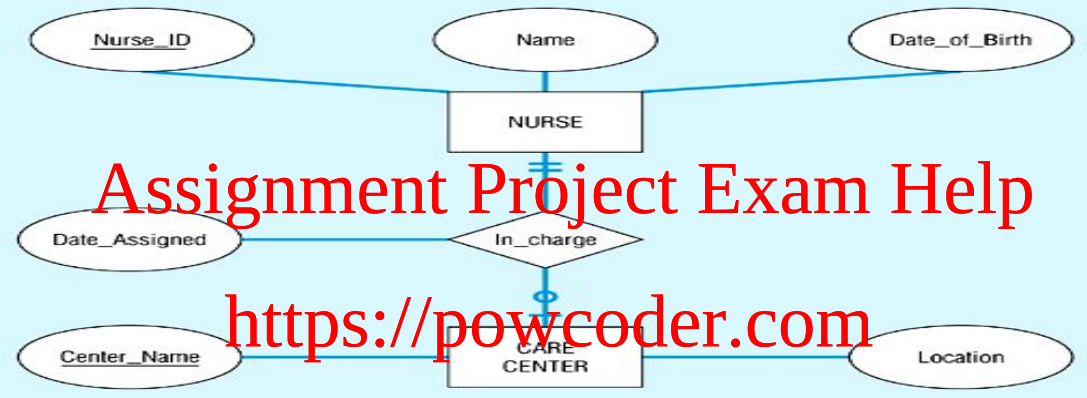
Map Binary Relationship (M:N)

- For a M:N binary relationship
  - First create a relation for each of the two entity types participating in the relationship.
  - Then create a new relation and include as foreign key attributes, the PK attribute (or attributes) for each of the two participating entity types. These attributes become the PK of the new relation.
  - If there are any nonkey attributes associated with the M:N relationship, they are also included in the new relation.

Mapping an associative entity with an Identifier



Mapping a 1:1 Binary Relationship



Relationship Participation
Mandatory vs Optional

NURSE participation *in this relationship*?

CARECENTRE participation *in this relationship*?

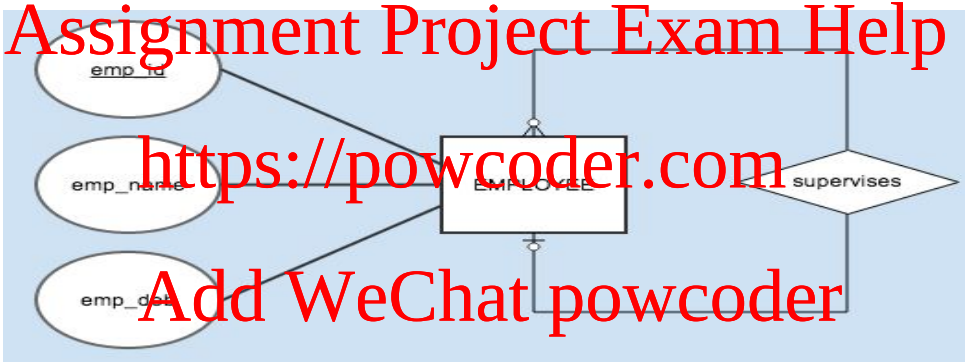
Map Binary Relationship (1:1)

- Create two relations, one for each of the participating entity types.
  - The primary key (PK) on the mandatory side of the relationship becomes the foreign key (FK) on the optional side of the relationship.
  - where both are optional place the FK on the side which causes the fewest nulls
  - Special case: 1:1 total relationship (mandatory participation from both sides)
    - Consider consolidating the two entity types into one relation

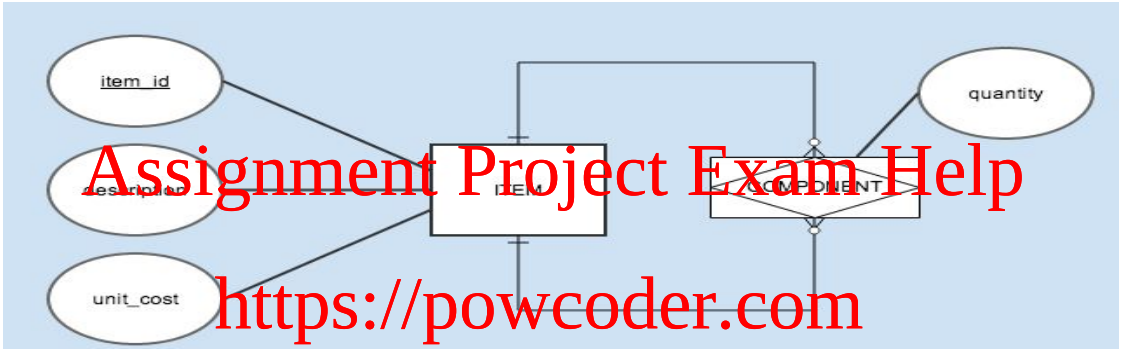
Map unary relationships

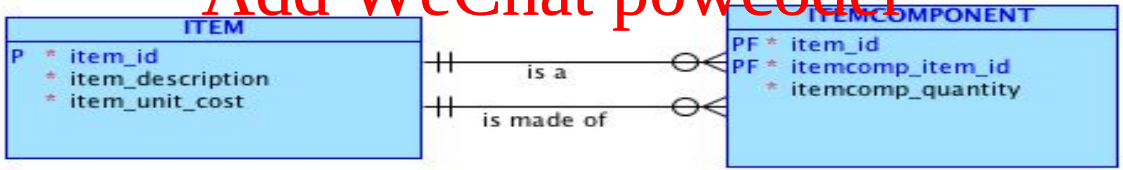
- Unary Relationship is a relationship between the instances of a single entity type.
- Unary 1:M Relationship – A relation is created for the entity type. Add a FK within the same relation that references the PK of the relation. A recursive foreign key is a FK in a relation that references the PK values of the same relation
- Unary M:N Relationship – Two relations are created, one for the entity type in the relationship and the other as the associative relation to represent the M:N relationship itself. The PK of the associative relation consists of two attributes (with different names) taking their values from the PK of the other relation.

Mapping a 1:M Unary Relationship

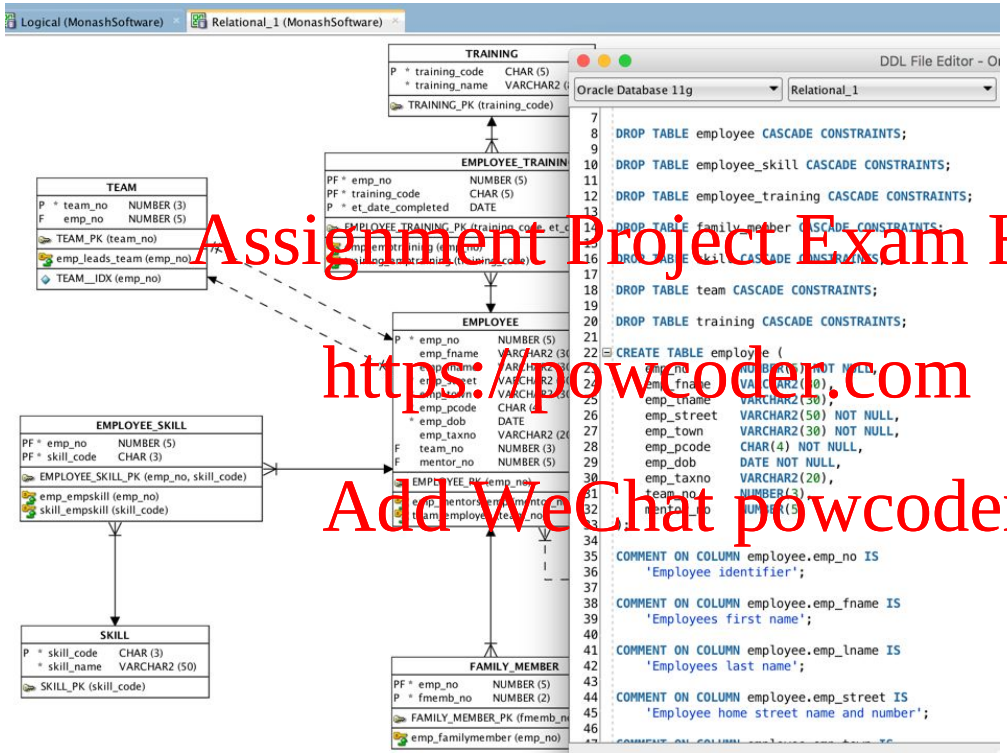


Mapping a M:N Unary Relationship

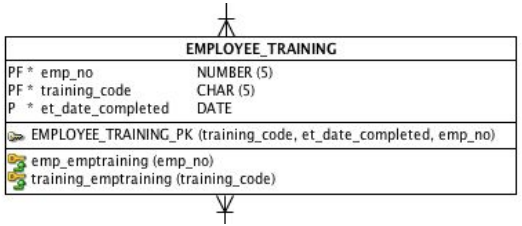




SQL Developer Data Modeler



Adding surrogate keys

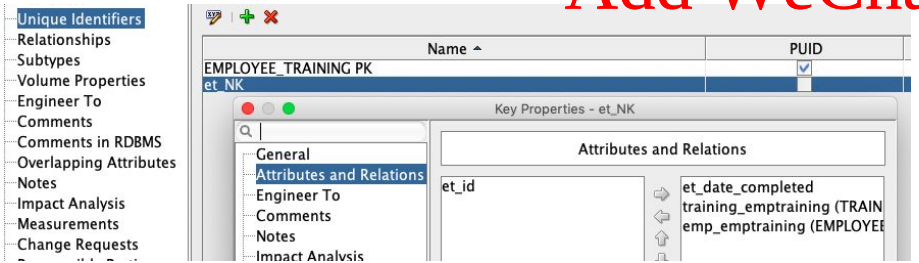


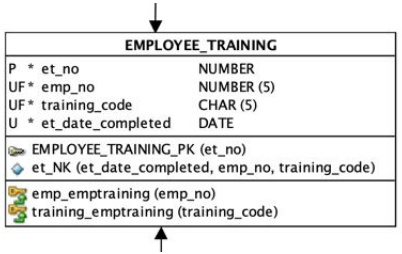
Potential problem:
Need to ensure that the identified key from the conceptual model - the natural key:
(emp_no, training_code, et_date_completed)
will still remain unique
Solution, where needed:
Define a unique index on the attributes of natural key

Surrogate PK's may be added **ONLY** on the logical model provided they are justified (include in documentation / assumptions).

MANUALLY add new PK attribute (here et_no), **DO NOT USE** SQL Developers "Create Surrogate Key" option

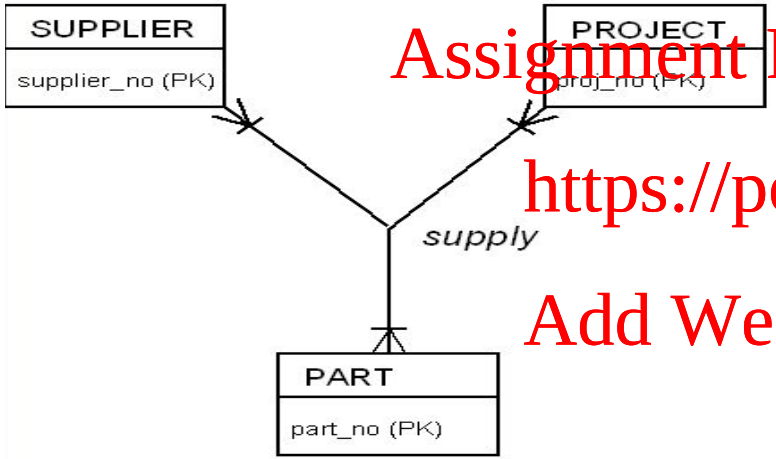
et_no	emp_no	training_code	et_date_completed
1	101	ORA01	1-Oct-2016
2	101	ORA01	1-Oct-2016
3	101	ORA01	1-Oct-2016



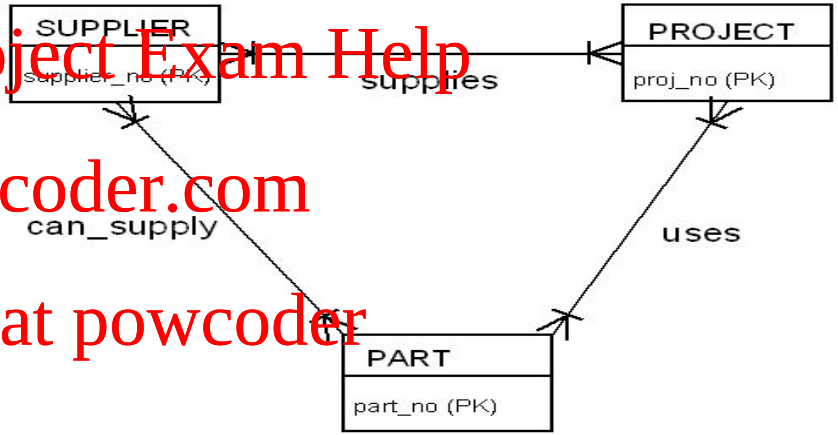


Ternary Relationships

Ternary



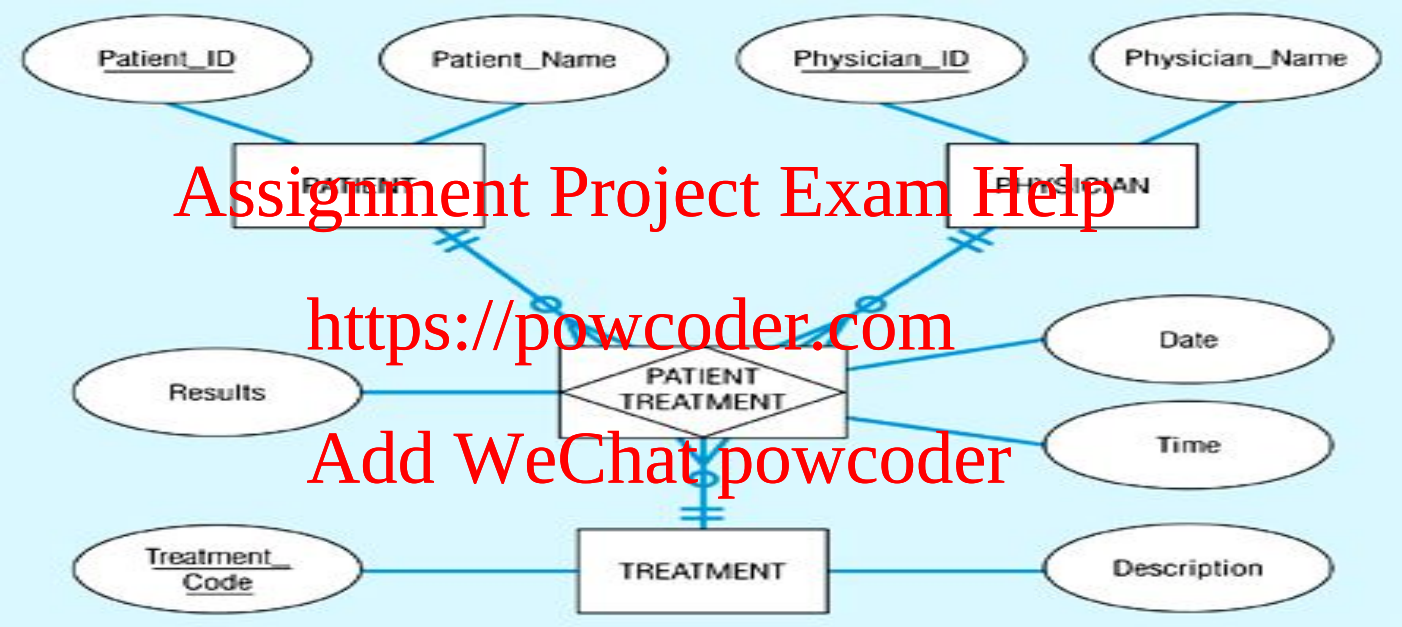
modelled as binary:



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
      - (supplier 1, project 2) (project 2, part 3) (supplier 1, part 3)
    - BUT does not imply (supplier 1, project 2, part 3)
- How then do we map such relationships?

Mapping a Ternary Relationship



Map Ternary (and n-ary) Relationships

- Ternary relationship should be converted to an associative entity.
  - To map an associative entity type that links three regular entity types, an associative relation is created.
  - The default PK of this relation consists of the three FK attributes for the participating entity types.
  - Any attributes of the associative entity type become attributes of the new relation.

Mapping a Ternary Relationship