

Johns Hopkins Engineering

Principles of Database Systems

Module #3

Conceptual Database Design II

IE (Crow's Foot) Notation

■ Relationship Cardinality

- Two pairs of (min, max) associated with Parent and Child Entities may vary due to business rules.



Optional One



Mandatory One



Optional Many



Mandatory Many

Exercise Examples:

- DEPARTMENT and EMPLOYEE
- EMPLOYEE and DEPENDENT
- EMPLOYEE and PAYCHECK
- EMPLOYEE and OFFICE

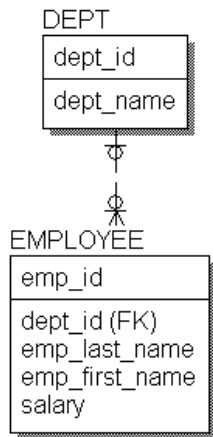
Creating ERD with A Database Design Tool

- Using database design tools (e.g. MySQL Workbench, Visio)
- Conceptual database design and logical database design:
 - Create entities and attributes
 - Create an ERD with DEPARTMENT, EMPLOYEE, and DEPENDENT entities, PKs, and non-key attributes
 - Create relationships
 - Identifying and non-identifying with FK migration
 - Can do "forward engineering " and/or "reverse engineering"
- Different tools may have different notations to draw ERDs. Read the tool documentation.

Creating 1:M with A Database Design Tool

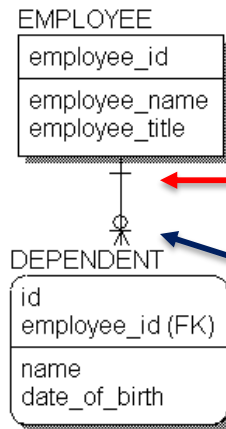
■ One to Many Relationships (IE using ERwin)

Non-Identifying Relationship




(Independent Entity)

Identifying Relationship



(Dependent Entity)

Note: “|” means (1, 1).

Note:  means (0, 1, M); that is equivalent to (0, M).

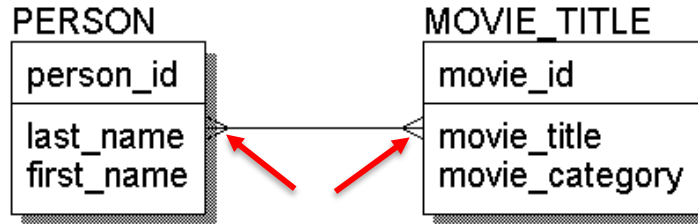
Two types of One to Many Relationships, identifying and non-identifying.

Parent to Child Rule: A DEPT contains many EMPLOYEEs.

Child to Parent Rule: An EMPLOYEE is associated with exactly one DEPT.

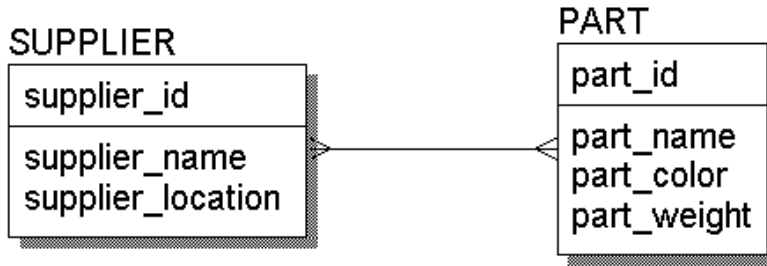
Creating M:N Relationship with A Database Design Tool

- Many to Many Relationships (IE using ERwin)



Note: A database design tool may not support many to many notation

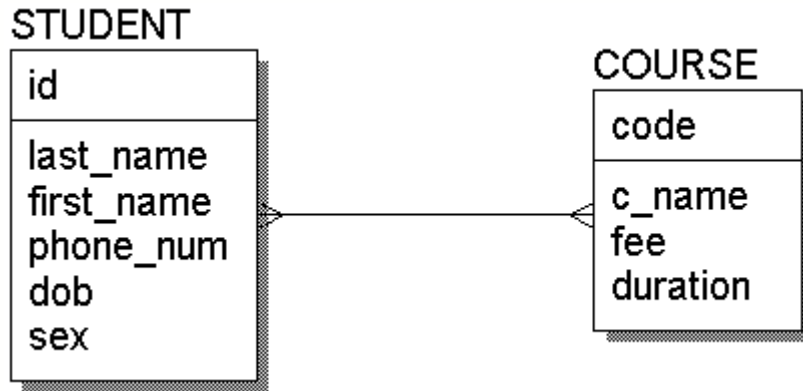
What's the date for "John Smith" to rent "Star Wars"?



How many of "Part 3" are supplied by "Supplier 1"?

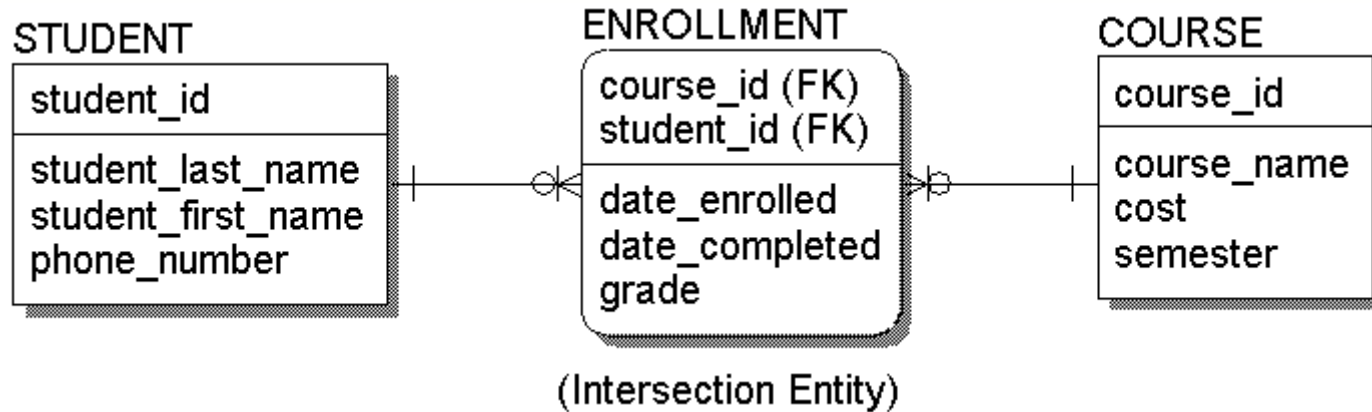
Creating M:N Relationship with A Database Design Tool (cont.)

- Attributes only describe entities
- Additional attributes are required to describe a relationship



Procedure to resolve M:M relationship

- Resolve the relationship by adding an intersection entity with two identifying relationships and attributes.



Procedure to resolve M:M relationship (cont.)

- In general, the UID of an Intersection Entity is composed of its relationships to the two originating entities.
- An Intersection Entity with no attributes is just a two-way cross-reference list between occurrences of the entities, e.g. supplier-sp-part relationship.
- The relationships from the Intersection Entity are normally mandatory.
- The relationships can be optional due to business rules, convenience, or performance.

Johns Hopkins Engineering

Principles of Database Systems

Module 3 / Lecture 2
Conceptual Database Design II

Common Approach to Develop ERD

- Identify the major entity types:
 - Identify the attributes for each entity type
 - Determine the unique identifier for each entity type
- Determine the relationships between the entity types:
 - Identifying or non-identifying (strong or weak entity types)
 - Mandatory or optional
 - Cardinality
- Resolve M:N relationships (not necessary to conceptual database design)
- Refine ERD as needed

Database Design Exercise

- Identify, and analyze the entities, the relationships, and attributes in the following set of information requirements:

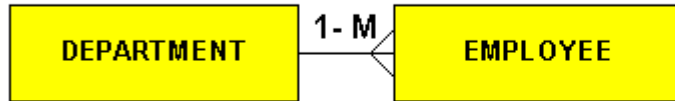
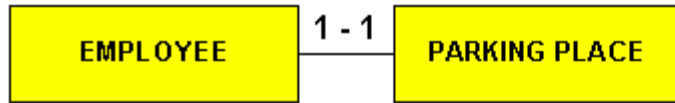
"I'm the manager of a training company that provides instructor-led courses in management techniques. We teach many courses, each of which has a code, a name, and a fee. Popular courses are Introduction to Network Security, Introduction to UNIX, Foundations of Algorithms, Software Project Management, XML: Technology and Applications, Foundations of Computer Architecture, and Introduction to TCP/IP. Courses vary in length from one day to five days. An instructor can teach several courses. New courses may be added based on students' requests or instructors' suggestions. Each course is taught by only one instructor. We create a course and then line up an instructor. We track each instructor's name and phone number. The students can take several courses over time, and many of them do this. We track each student's name and phone number. Some of our students and instructors do not give us their phone numbers."

Degree of a Relationship

- The number of entity types that participate in a relationship



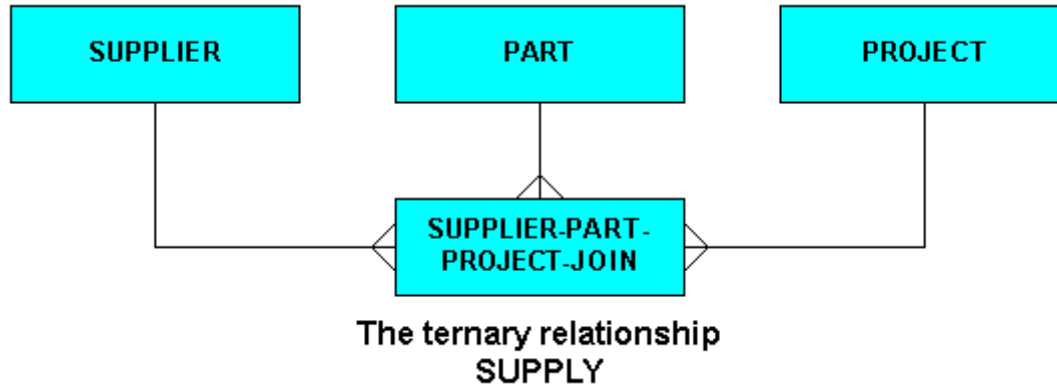
The unary relationships



The binary relationship

Degree of a Relationship (cont.)

- The number of entity types that participate in a relationship



Recursive Relationship

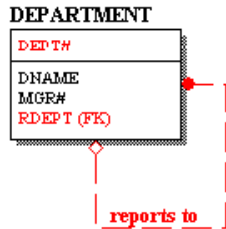
- A Recursive Relationship is a relationship between an entity and itself. In other words, a recursive relationship is one where the parent entity and the child entity for the relationship are the same entity.
 - Persons are parents of persons; employees manage employees; companies own companies.
 - This is an 1:M recursive relationship

Recursive Relationship (cont.)

- When a Primary Key (PK) attribute migrates from a parent entity to a child entity, it becomes a Foreign Key (FK) attribute in the child entity. Since a FK may have a different role of the related PK, a rolename is assigned to the attribute, e.g. emp_id, supervisor_id from EMP; place of birth and address with a reference from STATE entity.
- Recursive non-identifying relationships may be either mandatory (No Nulls) or optional (Nulls Allowed). However, in practice it is very unusual for a recursive relationship to be set to "No Nulls."

Recursive Relationship (cont.)

■ Examples (ERwin and MS Visio)



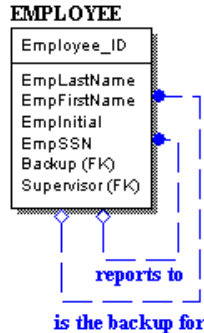
Entity-Attribute Editor Strings

Primary Key: DEPT#

Non-Key Attributes:

DNAME
MGR#
RDEPT.DEPT#

Note: RDEPT is a rolename.



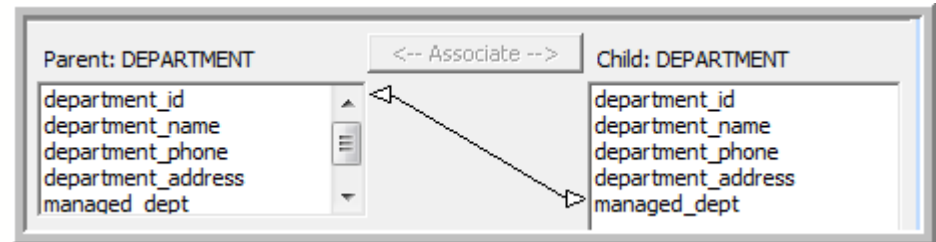
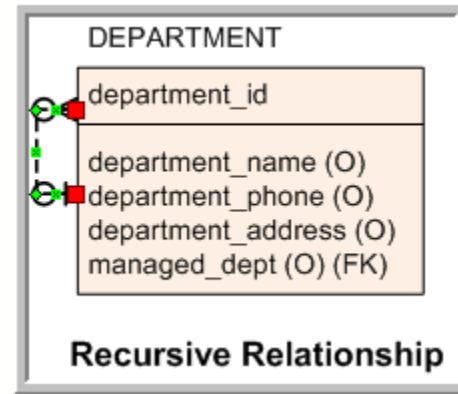
Entity-Attribute Editor Strings

Primary Key: Employee_ID

Non-Key Attributes:

EmpLastName
EmpFirstName
EmpInitial
EmpSSN

Note: Backup and Supervisor attributes are references to Employee_ID

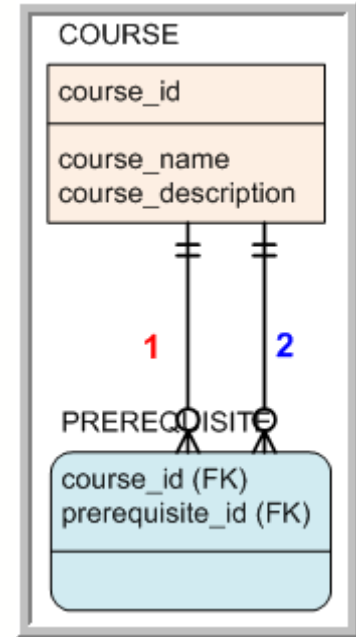


M:N Recursive Relationship

- An M:N recursive relationship:
 - One course has zero, one, or many prerequisites. The course may be other courses' prerequisites.
 - One part contains many other parts, and the part can be a component of many other parts.
- Like an M:N relationship, this can be resolved into an intersection entity with two 1:M relationships.

M:N Recursive Relationship (cont.)

- Example: Course and it's prerequisites
 - **Relationship 1:** A course has zero to many prerequisites. COURSE.course_id equals PREREQUISITE.course_id as a FK
 - **Relationship 2:** A course may be other courses' prerequisites. COURSE.course_id equals PREREQUISITE.prerequisite_id as a FK
 - Additional constraint:
PREREQUISITE.prerequisite_id is not equal to PREREQUISITE.course_id



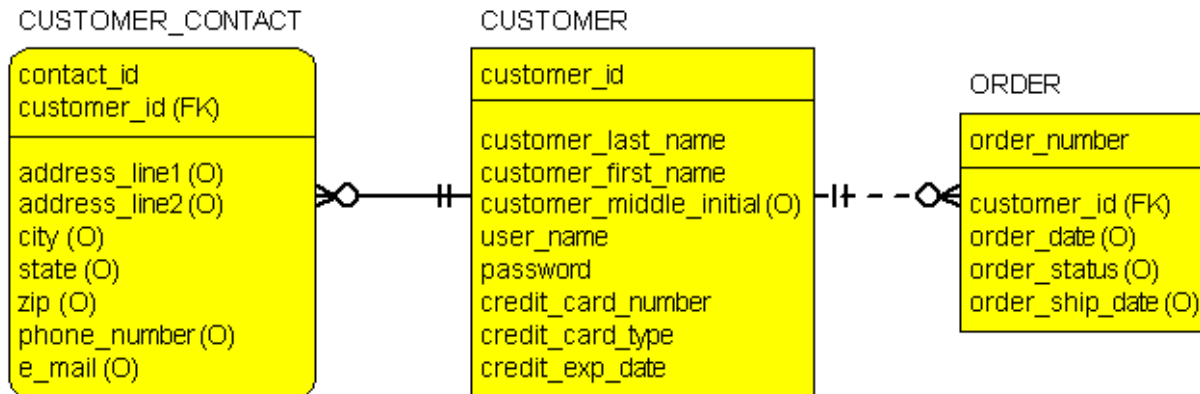
1:1, 1:M and M:N Relationships

- The **cardinality ratios** and **participation constraints** together are the structural constraints of a relationship type.
- A relationship can be identifying or non-identifying.
- How to determine identifying or non-identifying relationships:
 - From Child Entity Type to Parent Entity Type
 - Does the Child Entity Type have a key attribute?
 - Child is **DEPENDENT** upon the Parent for its identification and cannot exist without the parent

1:1, 1:M and M:N Relationships (cont.)

■ Examples:

- Each CUSTOMER places zero to many ORDERs
- Each ORDER is received from one and only one CUSTOMER
- An ORDER can be identified without information about a CUSTOMER, but requires a value for customer_id (mandatory and not null)
- An CUSTOMER_CONTACT requires customer_id or customer's info (e.g., name) in order to get the proper CUSTOMER's contact information



Johns Hopkins Engineering


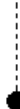

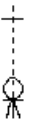





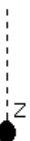










Principles of Database Systems

Module 3 / Lecture 3
Conceptual Database Design II

Summary of Relationship / Cardinality Symbols in IE Notation:

Know your database design tool notation

Summary of Relationship/Cardinality Symbols in IE Notation

Cardinality Description	IDEF1X Notation		IE Notation	
	Identifying	Non-identifying	Identifying	Non-identifying
One to zero, one, or more				
One to one or more				
One to zero or one				
Zero or one to zero, one, or more (non-identifying only)				
				
Zero or one to zero or one (non-identifying only)				
				

Know your database design tool notation

Converting an ERD from Chen's Notation to IE Notation

- **Strategy: Divide and Conquer**
- **Strong Entity vs. Weak Entity**
- **Identifying vs. Non-identifying Relationships;**
- **Total Participation (Existence Dependence) vs. Partial Participation;**
- **Cardinality Ratios;**
- **Foreign Key Migration;**
- **M:N Relationship Resolution**

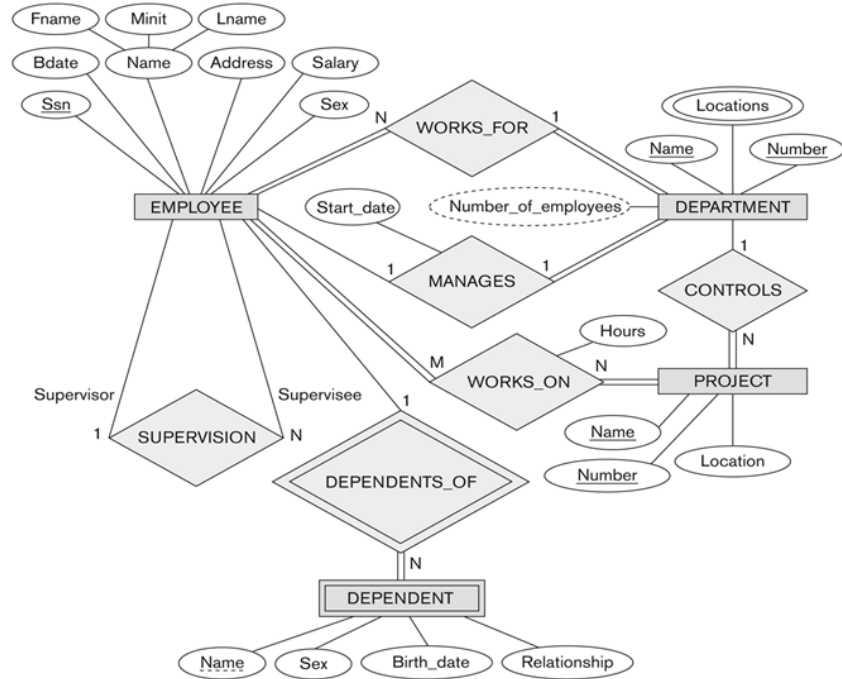
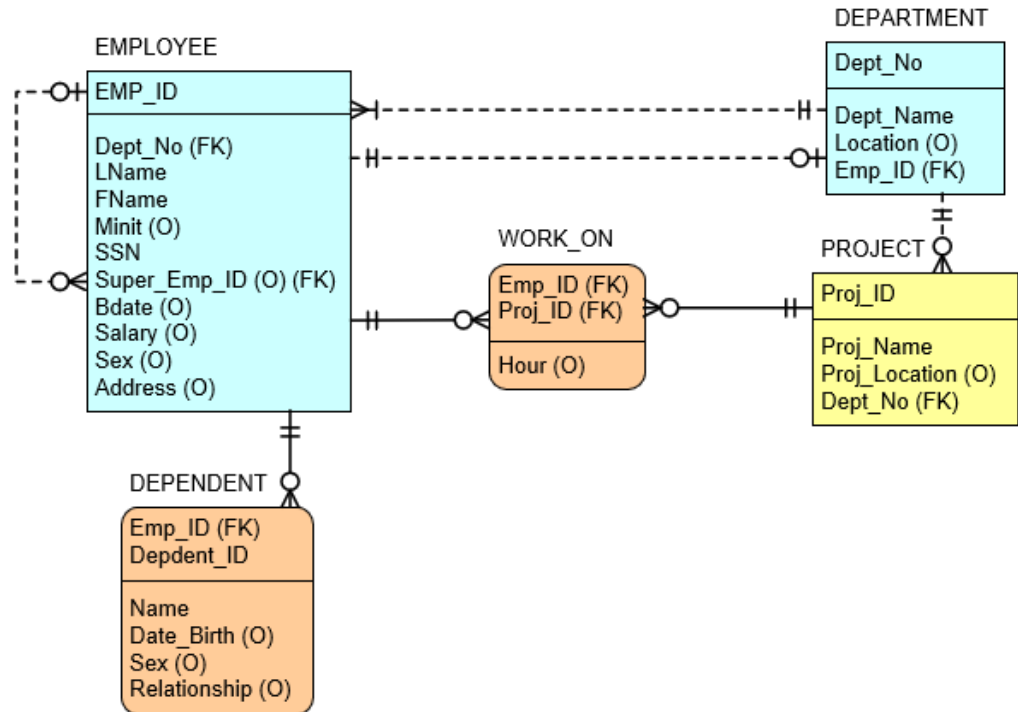


Figure 3.2
An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.

Converting an ERD from Chen's Notation to IE Notation (cont.)

- **Strategy: Divide and Conquer**
- **Strong Entity vs. Weak Entity**
- **Identifying vs. Non-identifying Relationships;**
- **Total Participation (Existence Dependence) vs. Partial Participation;**
- **Cardinality Ratios;**
- **Foreign Key Migration;**
- **M:N Relationship Resolution**



Company ERD Review

- Design questions on Company ERD:
 - Can an employee exist without assigning a department who they can work for?
 - Can a department exist without assigning an (or four) employee?
 - Can a department exist without assigning a department head?
 - Can a project exist without assigning a controlling department?
 - Can a dependent exist without associating an employee?
 - Business rules for each company may be different. What are reasonable business rules – a general consensus?

Supertype and Subtype Relationship

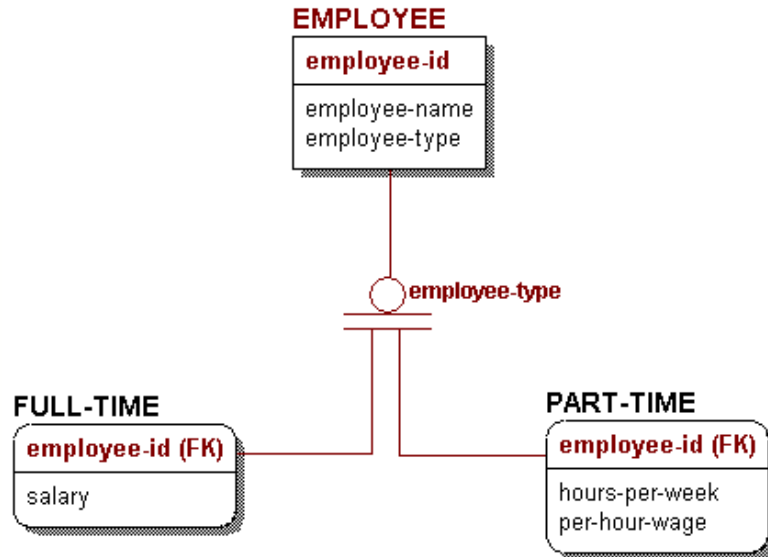
- A subtype relationship (also known as a categorization relationship) is a relationship between a subtype entity and its generic parent. A subtype relationship always relates to one instance of a generic parent with zero or one instance of the subtype.
- A COMPANY should be either a SUPPLIER or a BUYER. Both SUPPLIER and BUYER must be only one COMPANY and are examples of subtype entities while the COMPANY is supertype.
- In general, this is a mandatory one-to-one relationship.

Supertype and Subtype Relationship (cont.)

- The relationship can be complete or incomplete.

Example of Supertype and Subtype

Note: PERSON-NAME is a DOMAIN name



Category: complete/incomplete
Parent to category
Category to child

Category discriminator
employee type: full/part time
employee class: secretary...

Arc Relationship

- Oracle designer uses Arc relationship for this exclusive OR case while ERwin uses supertype and subtype relationships.
- For example, a MEMBERSHIP must be held by either a CUSTOMER or a COMPANY. A CUSTOMER or a COMPANY may each hold more than one MEMBERSHIP.

Relationship Reading Syntax

- Each source entity (may be / must be) relationship name (one and only one / one or more) destination entity.
- Each direction of a relationship has a name, e.g. taught by or assigned to; an optionality (must be / may be), a degree (one and only one / one or more).

Naming Convention

- Based on requirements, a noun will be either an entity type, or attribute (or an instance). The verbs usually indicate names of relationship types.
- Use upper case for an entity type name and lower case for an attribute. Name should be descriptive. Entity type name is singular.
- Organize the E/R diagram with top-down and left-to-right, grouping, and coloring to increase readability, and to support the business operations.

Modeling Business Rules using ER Diagram

- Entity types with attributes and relationships
- Categories of sets of data within data
- Mandatory versus optional relationship
- Multiple relationships

Data Modeling and Database Design

- Create a business narrative based on interview notes, requirements, and specifications.
- Create an ER diagram. ER diagrams are derived from business narratives.
- Create a database design. In a relational database, the ER diagram maps entity types to tables, attributes to columns, relationships to foreign keys, and business rules to constraints.
- Create a table definition. The basis for the SQL CREATE TABLE statement creates tables, columns, and constraints.

Advantages for using ER Model

- Easy presentation for the design
- Easy to communicate with users
- Easy to be developed, and refined

The ER diagram is a graphical representation of business information needs and rules.

Any change made during later stages of the development life-cycle can be very expensive. Therefore, the conceptual modeling stage is very important for properly implementing the system requirements.

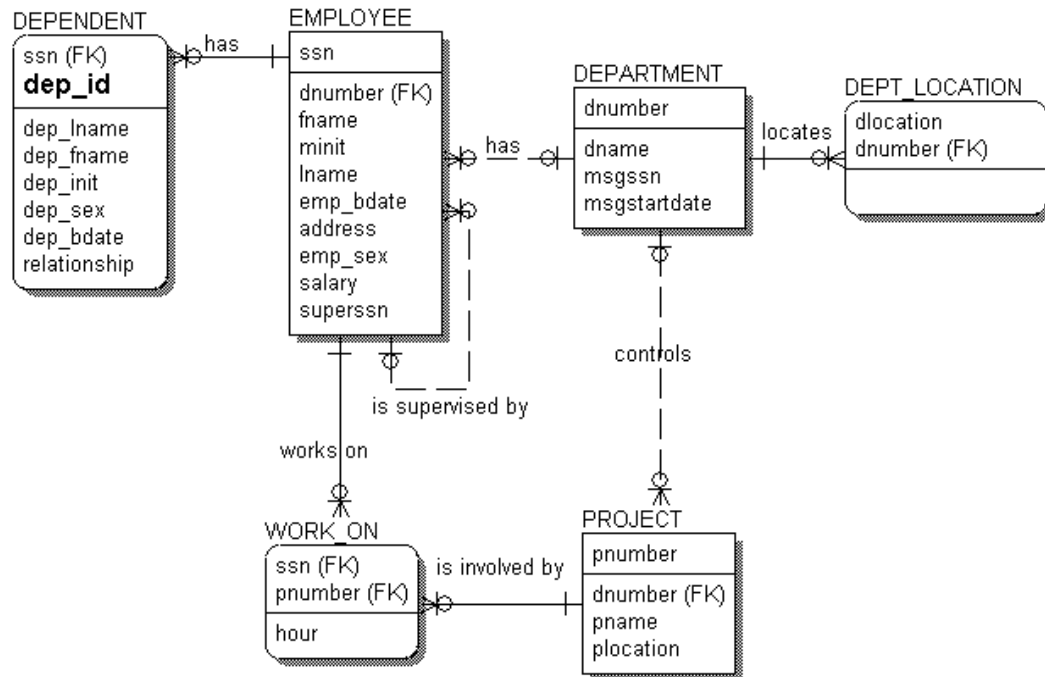
Johns Hopkins Engineering

Principles of Database Systems

Module 3 / Lecture 4
Conceptual Database Design II

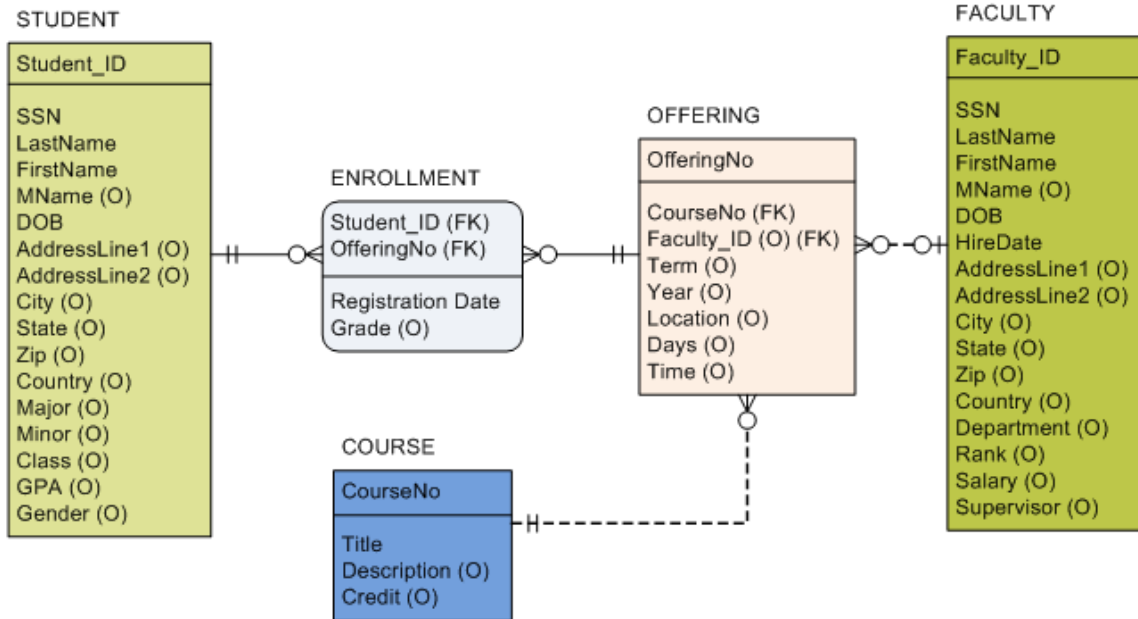
Company ERD Using ERwin

ER Diagram for the COMPANY



Different Database Design tools may have slightly different notations to create ERDs.

Case Study – University ERD



Questions:

- Explain how to model Term in the OFFERING table
- Explain how to model faculty rank in the FACULTY table
- Explain how to model Country attribute in the STUDENT table and FACULTY table
- Explain how to model department in the FACULTY table
- Explain how to model Student Majors and Minors
- Explain how to model that a student is also a faculty member
- Discuss the pro and con if a PERSON table to cover both STUDENT and FACULTY

Form Analysis

- Requirements may come from a form, a written document, or both.
- Steps in the form analysis process -
 1. Define and Analyze Form Structure:
 - Construct a hierarchy from the form structure. Most forms consist of a simple hierarchy where the main form is the parent and the subform is the child. Complex forms (subforms inside subforms) are not as common because they can be difficult for users to understand.

Form Analysis (cont.)

- Steps in the form analysis process -

- 2. Identify Entity Types:

- Map and split the hierarchical structure into one or more entity types.
 - Look for form fields that can be a primary key(s) of an entity type.
 - Group form fields into entity types using functional dependencies.

- 3. Attach Attributes:

- Attach attributes to the entity types identified in the previous step.
 - Be aware of attributes belonging to a Many-to-Many relationship that require an additional entity type.

Form Analysis (cont.)

- Steps in the form analysis process -

- 4. Add Relationships:

- Connect entity types with (identifying or non-identifying) relationships and specify cardinalities.

- 5. Check Completeness and Consistency:

- Check the ERD for consistency and completeness with the form structure
 - Verify minimum and maximum cardinalities for all relationships, a primary key for all entity types, and a name for all relationships.

Form Analysis: Invoice Example

Parent Form

Vendor Information

- Vendor Name
- Vendor Address

Order Information

- Order No
- Order Date

Customer Information

- Customer No
- Customer Name
- Customer Address

- SalesPerson No
- SalesPerson Name

Invoice Information

- Invoice No
- Invoice Date

Ship Information

- Ship Date
- Shipping Service

Child Form

Product Information

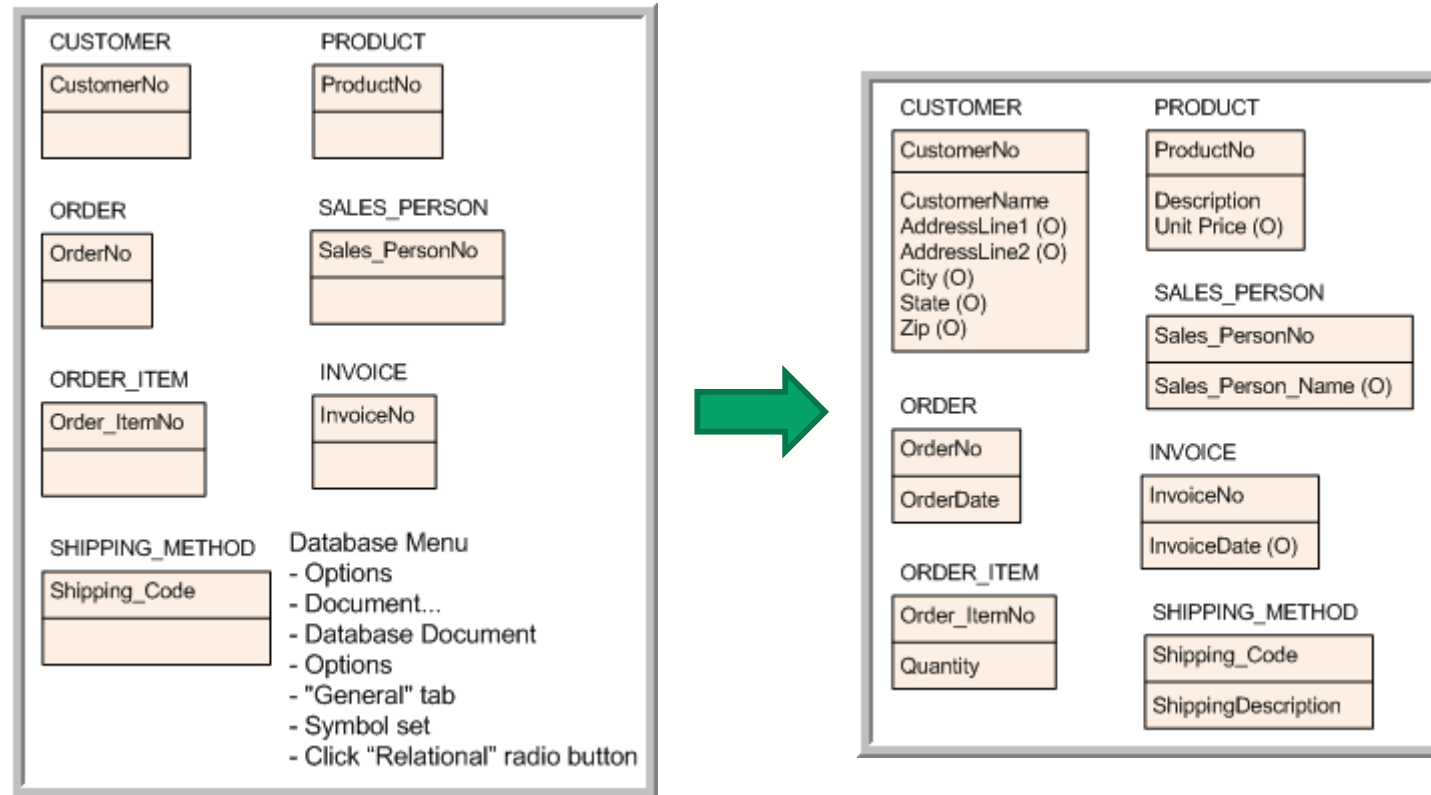
- Product No
- Product Description
- Quantity
- Unit Price
- Total

- Subtotal
- Freight
- Tax
- Total Cost

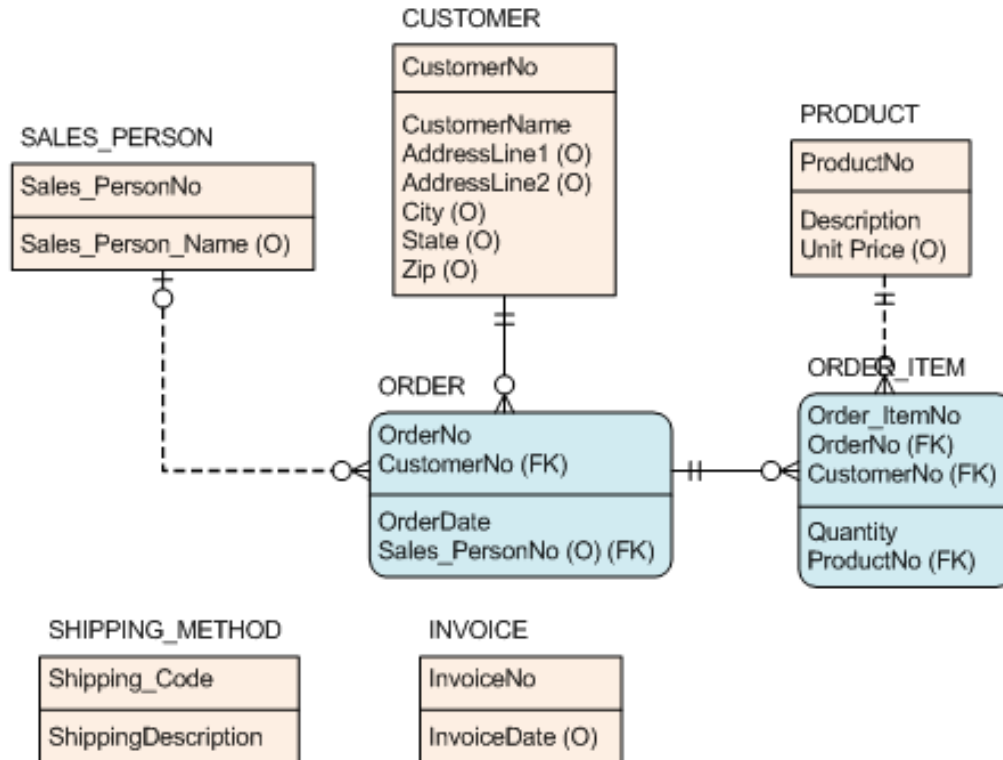
Parent Form includes entities with one occurrence on a form.

Child Form may include multiple occurrences on a form.

Form Analysis: Invoice Example (cont.)



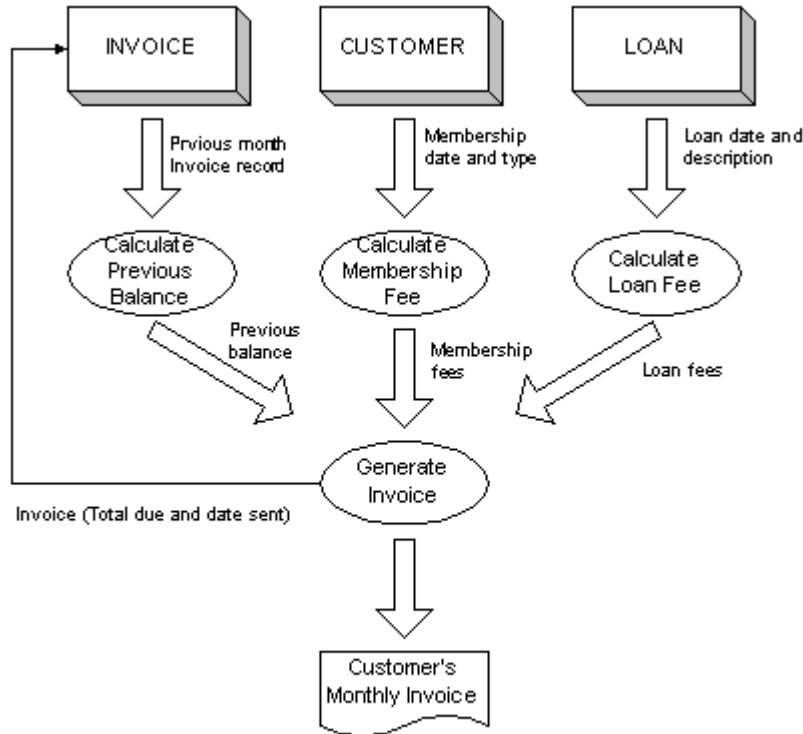
Form Analysis: Invoice Example (cont.)



Questions:

- Do you need a COMPANY for vendor information?
- What is the relationship between ORDER and INVOICE entities?
- What is the role for SHIPPING METHOD entity?
- How do you handle BILL TO and SHIP TO information?
- How to enforce the STATE reference occurred at various entities?

Form Analysis: Invoice Example (cont.)



Invoice Processing:

The invoice process may involve multiple entities and calculations in order to generate a Customer's Monthly Invoice.

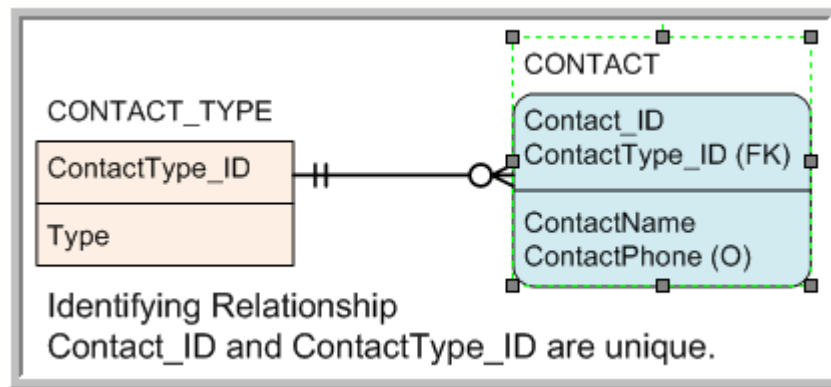
Johns Hopkins Engineering

Principles of Database Systems

Module 3 / Lecture 5
Conceptual Database Design II

An Alternative Design for An Identifying Relationship

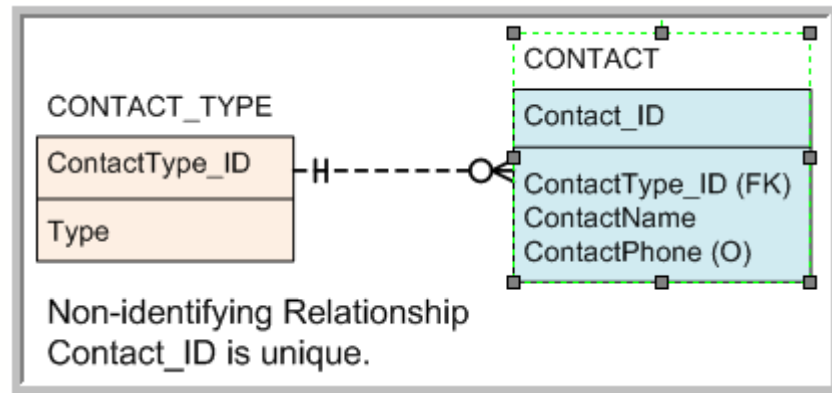
- Using an identifying relationship, a PK of a parent entity will migrate to a child entity as a FK as well as a part of PK.
- The child entity may have a composite PK with many attributes that can be an overhead on implementation.



	Physical Name	Data Type	Req'd	PK
►	Contact_ID	INTEGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ContactType_ID	INTEGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ContactName	VARCHAR(30)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	ContactPhone	CHAR(10)	<input type="checkbox"/>	<input type="checkbox"/>

An Alternative Design for An Identifying Relationship (cont.)

- The alternative design with an identifying relationship has the following properties:
 - The child entity becomes a strong entity.
 - The PK of the child entity remains unique and NOT NULL with an artificial sequence number.
 - The FK remains FK with a NOT NULL constraint.



	Physical Name	Data Type	Req'd	PK
►	Contact_ID	INTEGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	ContactType_ID	INTEGER	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	ContactName	VARCHAR(30)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	ContactPhone	CHAR(10)	<input type="checkbox"/>	<input type="checkbox"/>

Checking the Data Model

- The check process is to ensure that the information required by the functions can actually be derived from the model.
 - Make sure entity types are connected properly so that the required data can be effectively retrieved from the database.
 - Look into the functional usage of the data.
 - Check 1:M identifying relationships and non-identifying relationships.
 - Check M:N relationships and resolve with intersection entity types.
 - Check recursive relationships (normally non-identifying relationship).

Relevant Database-related Tools

- Comparison of RDBMSs - Compare general and technical information OS support, Limits, Partitions, DB capabilities:
http://en.wikipedia.org/wiki/Comparison_of_relational_database_management_systems#Fundamental_features
- Comparison of Database Administrator Tools - Compare general and technical information:
http://en.wikipedia.org/wiki/Comparison_of_database_tools
- Web Application Development Tools - the process and practice of developing web applications:
https://en.wikipedia.org/wiki/Web_application_development

Making A Decision on Technology Solution

- Best technology or most popular technology (e.g. Borland Delphi)
- Cost
- Future vendor perspectives:
 - Company size and financial condition
 - Company merge or acquisition
 - Oracle bought BEA, Sun Microsystems, PeopleSoft, J.D Edward, Siebel and others
 - SAP acquired Sybase
 - Microsoft
 - IBM brought Informix

Conceptual Database Design Conclusions

- Good data modeling:
 - Lead a model/structure to be smaller, faster, simpler, more understandable, and easier to maintain.
 - Decrease development time.
 - Increase database system performance and quality.
- A good model comes from skilled staff with intelligence on abstract and analytical thinking, experience, and communication skills.
- If possible, use commercial database design tools to create your model.