Module 2.3 – Translate EER Models to Relations

2. DB Modeling

- 2.1 Entity Relationship (ER) Model
  - Entities, Relationships, Constraints
- 2.2 Extended Entity Relationship (EER) Model
  - Specialization, Subtype, Supertype, Subtype Discriminator
- 2.3 Translation of EER Models to Relations

2.1 ER Model

Learning Outcomes

- EERM
  - Understand and Explain Extended ER modeling concepts
    - Inheritance, Subtypes, supertypes, specialization hierarchy, subtype discriminator; Overlapping vs disjoint, mandatory vs optional completeness
  - Understand that various standards are used in ER modeling (Chen, **Crow's foot**, UML, …)
  - Explain and use Associative entities
  - Create ER diagrams from user requirements
- Selecting Primary Keys & Flexible Design
  - Understand the selection process for primary keys and when to use surrogate keys
  - Explain: implementing 1:1 relationships, fan-traps, redundant relationships
  - Design for maintaining historical data

Textbook Readings

- Chap 5

Testing*

*Main (but not the only ones) sections of the textbook used for testing are identified in parentheses

- Entity Relationship modeling (5.1, 5.3, 5.4)

Logical Design

EER model needs to be translated into the relational model:

1. Represent entities
2. Represent relationships
3. Normalize the relations
4. Merge relations if required

We cover the first two steps. The last two steps will be covered later in the course.

Translation of ER Model into a DB Model

Translating EER diagrams into DB tables

1. Represent entities … table for each entity
   - Attributes (”regular”, composite, multi-valued, identifying)
   - Weak entities
   - Sub-types
2. Represent relationships
   - One-to-One
   - One-to-Many
   - Many-to-Many
3. Represent subtypes
4. Document all business rules represented in the ER diagram
   - Business rules need to be enforced/followed
     - Rules that are enforced by DBMS due to DB schema/structure
     - Rules that are not enforced by DBMS due to DB schema … need to be enforced by scripts

Represent Entities

Create a table for each entity

- Single-valued attribute -> table attribute
- Composite attribute
  - Individual attributes, of a composite attribute, become attributes of the table
  - Applied recursively (if a composite attribute contains a composite attribute)
- Entity identifying attributes
  - Corresponding table attributes/columns must have unique values and one is chosen as primary keys (careful if identifying attributes are composite)
  - If none exist => Weak entity?
- Multi-valued attribute A of the Entity E with its corresponding table R
  - Create a new table containing the multi-valued attribute and the foreign key being the primary key of R (new table has the foreign key (primary key (could be composite)) of R plus the multivalued attribute A

Represent Weak Entities and Subtypes

Create a table for each entity … continued

- …
- Create a new table for each weak entity
  - Weak entity is always in a dependency relationship with a strong (regular/normal) entity
  - The primary key of the weak entity is a composite key consisting of the primary key of the strong entity and the weak entity's identifying attribute
    - **If a weak entity does not have an identifying attribute (that, in conjunction with the primary key of the strong entity, would form a primary key, add a surrogate key to the new table and use it .**
- Create a new table for each sub-type
  - with the attributes of the sub-type forming the attributes of the new table table
  - primary key of the table representing the parent entity is added to the new table – it becomes the table's primary key (and is a foreign key)

Represent Entities: Identifying & Composite Attributes

Identifying Attribute:

One of the identifying attributes is chosen as the primary key

Composite Attribute:

If Address were a composite attribute consisting of:

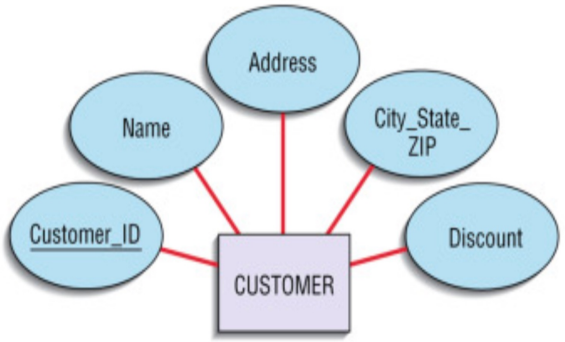
St. No, Street, City

Customer table would contain attributes:

St no., Street, City

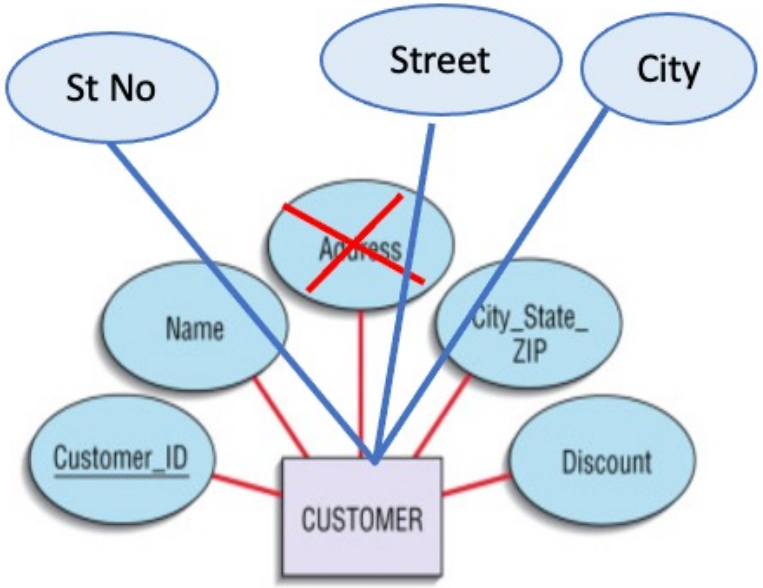
Instead of the attribute Address

Example:



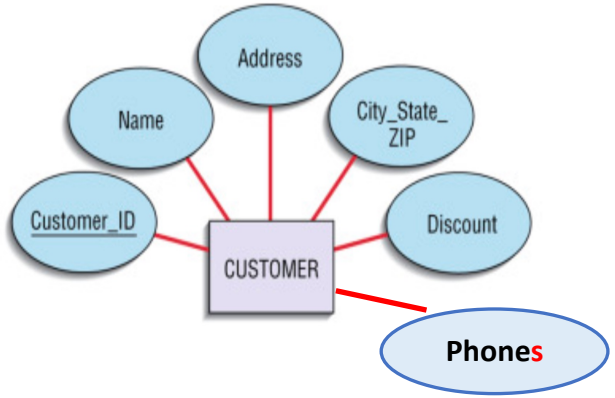
CUSTOMER

Customer_ID	Name	Address	City_State_ZIP	Discount
1273	Contemporary Designs	123 Oak St.	Austin, TX 28384	5%
6390	Casual Corner	18 Hoosier Dr.	Bloomington, IN 45821	3%



Represent Entities: Multivalued Attribute

Example:



- If there were a multi-valued attribute *Phone* in *Customer* entity
  - Create new table (e.g., called *Phones*) with attributes
    - Primary key of table *Customer* (as a foreign key)
    - Attribute *Phone*
    - Primary key is the composite attribute consisting of all attributes (*Customer_ID, Phone*)

CUSTOMER

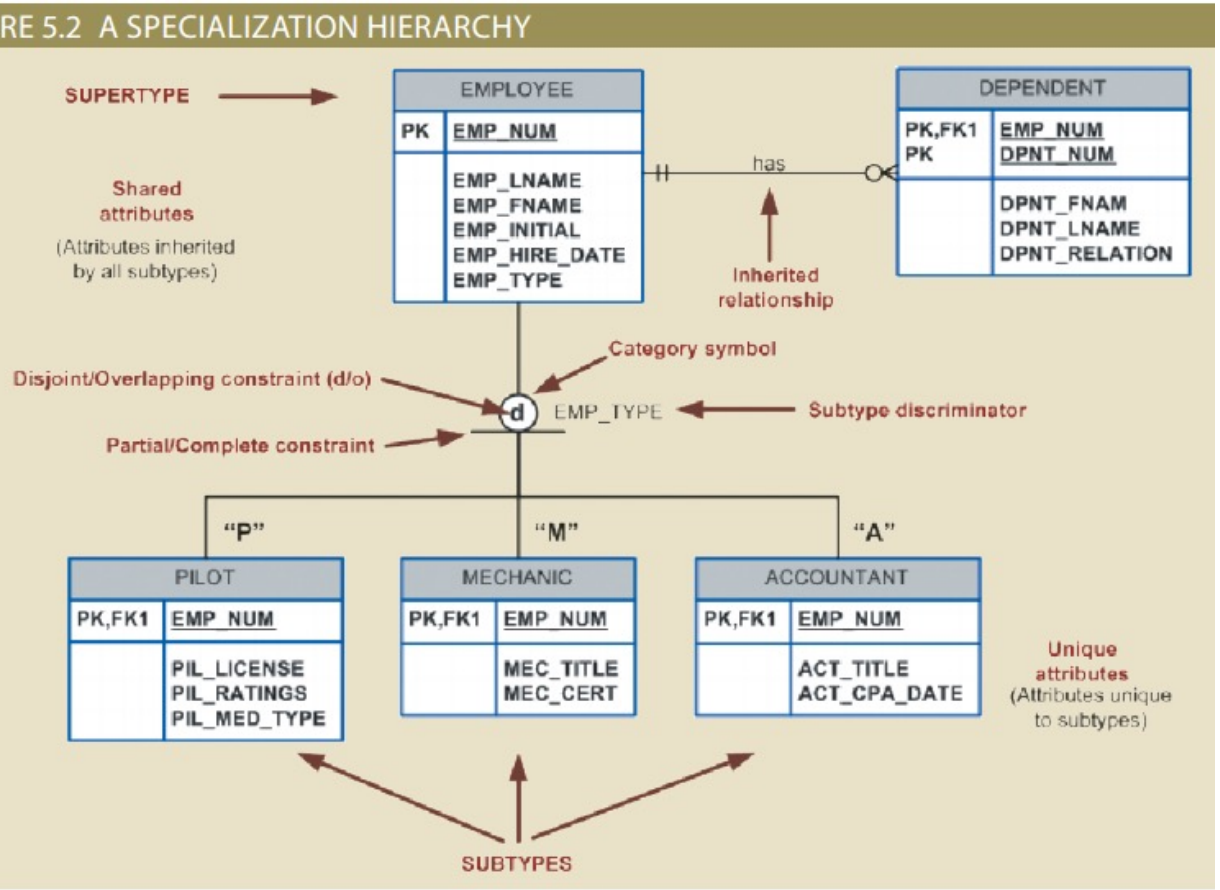
Customer_ID	Name	Address	City_State_ZIP	Discount
1273	Contemporary Designs	123 Oak St.	Austin, TX 28384	5%
6390	Casual Corner	18 Hoosier Dr.	Bloomington, IN 45821	3%

CUSTOMER_PHONES

Customer_ID	Phone
1237665	902-123-4567
.....	.....
1237665	902-123-4568

Represent Subtypes

Example:



New table for each sub-type

- with attributes of the subtype
- plus the with the primary key of parent
  - is a *foreign key* and serves as the *primary key:*
    - PILOT (EMP_NUM (PK, FK), PIL_LICENCE, PIL_RATINGS, PIL_MED_TYPE)
    - MECHANIC (EMP_NUM (PK, FK), MEC_TITLE, MEC_CERT)
    - MECHANIC (EMP_NUM (PK, FK), ACT_TITLE, ACT_CPA_DATE)
- Note that in the diagram, the key EMP_NUM is added by the diagraming tool when a subtype is s created

Represent Weak Entities

Example:

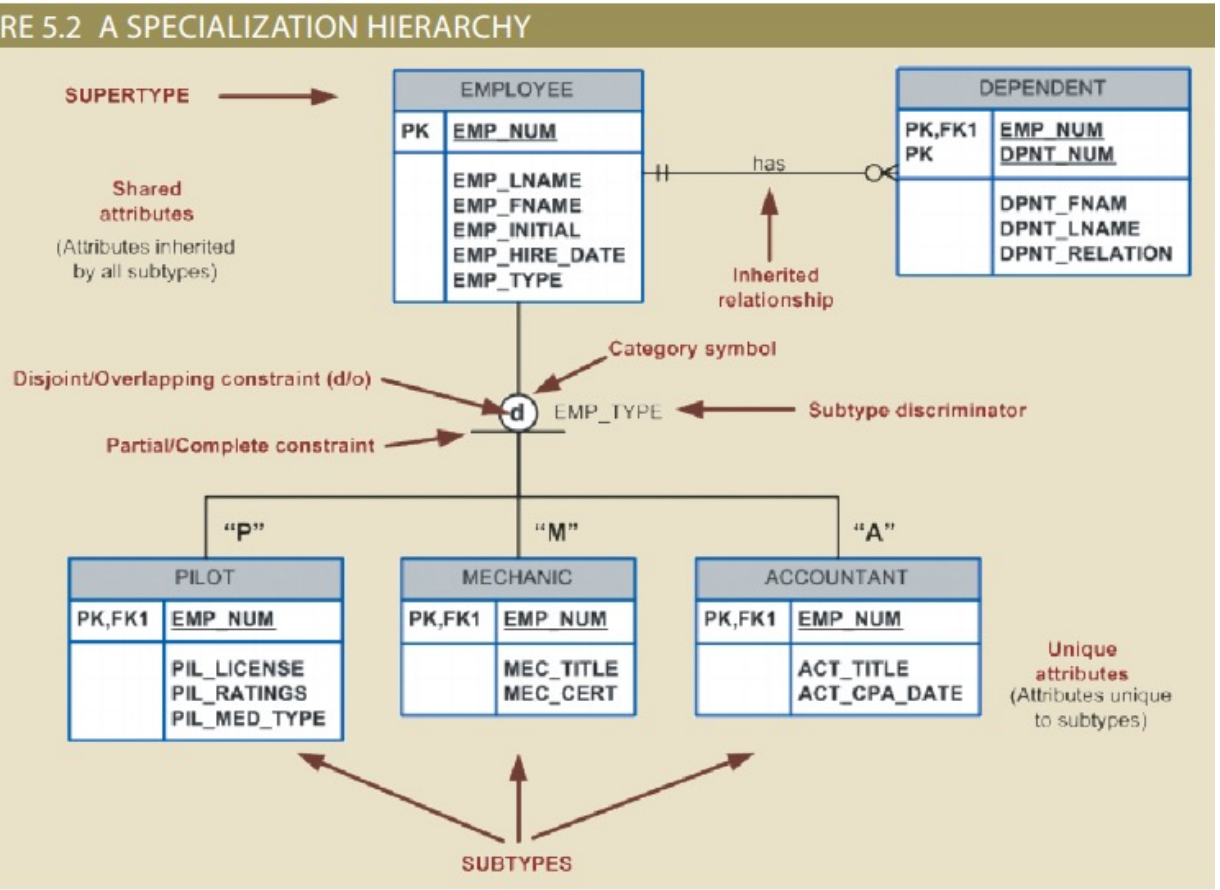


Table for a weak entity

DEPENDANT (

DEPT_NUM (PK),

DEPT_FNAME, DEPT_LNAME, DEPT_RELATION)

Note that the key EMP_NUM is added by the diagraming tool when the relationship is created

Represent Relationships

Entities A, B, and C are represented by tables X, Y, and Z, respectively.

- One-to-one relationship between A and B
  - Include/add primary key of X in table Y or include/add primary key of Y into X.
  - Which option? Choose option that leads to the smallest number of NULL values.

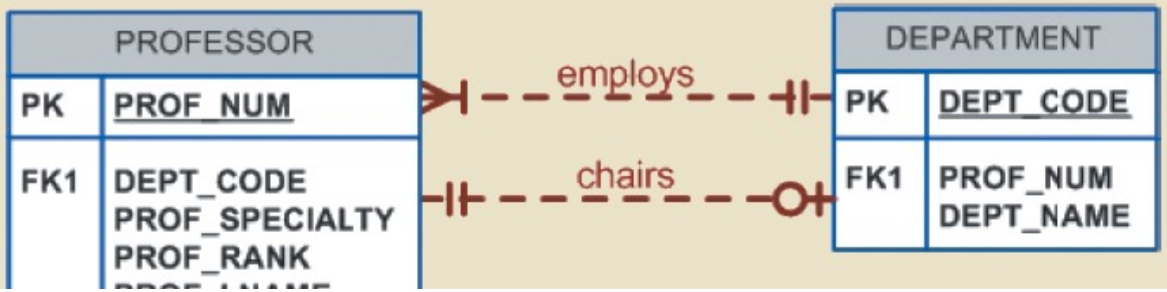
- One-to-many relationship from A (“on the one-side of the relationship”) to B (“on the many-side of the relationship
  - Adding primary key from table representing the “one-side” of the relationship, as foreign key, into the table of “many-side” of the relationship

- Many-to-many relationship between A and B
  - Create a new table that has attributes that are the primary keys of X and Y, which serve as foreign keys, and together they form the composite primary key of the new table.
  - If the relationship has any attributes, they are also added to this table

Represent Relationships – One-to-One

Example:

FIGURE 4.29 THE FOURTH TINY COLLEGE ERD SEGMENT



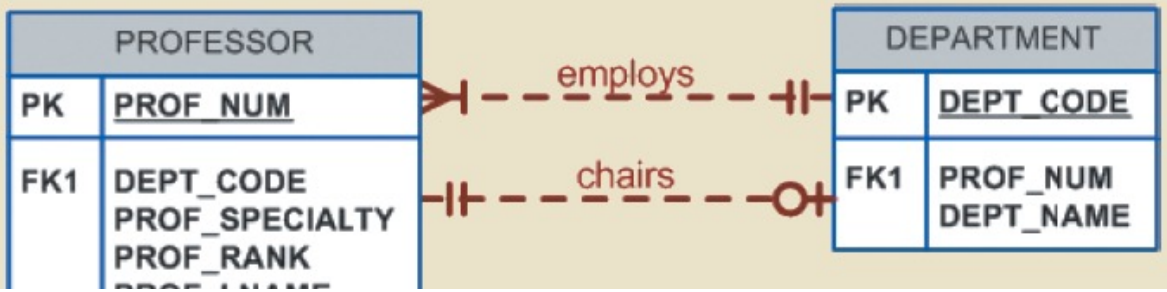
One-to-One relationship *chairs*

- Already have tables PROFESSOR and DEPARTMENT
- Options
  - Add PROF_NUM to table DEPARTMENT
  - Add DEPT_CODE to PROFESSOR
- Choose - **Add PROF_NUM to table DEPARTMENT** as it results in the smaller number of NULL values
- Note that MySQL Workbench adds the key automatically when the relationship is created

Represent Relationships – One-to-Many

Example:

FIGURE 4.29 THE FOURTH TINY COLLEGE ERD SEGMENT



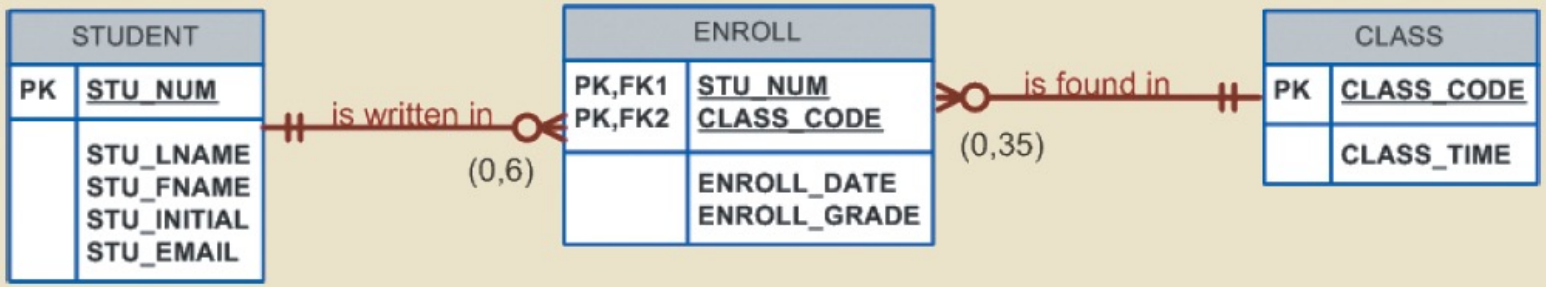
One-to-Many relationship *employs*

- Already have tables PROFESSOR and DEPARTMENT
- Add **DEPT_NUM** from DEPARTMENT to table PROFESSOR
- Table PROFESSOR has DEPT_CODE as a foreign key
- Note that MySQL Workbench adds PROF_NUM automatically when the relationship is created for an identifying relationship

Represent Relationships – Many-to-Many

Example:

FIGURE 4.31 THE SIXTH TINY COLLEGE ERD SEGMENT

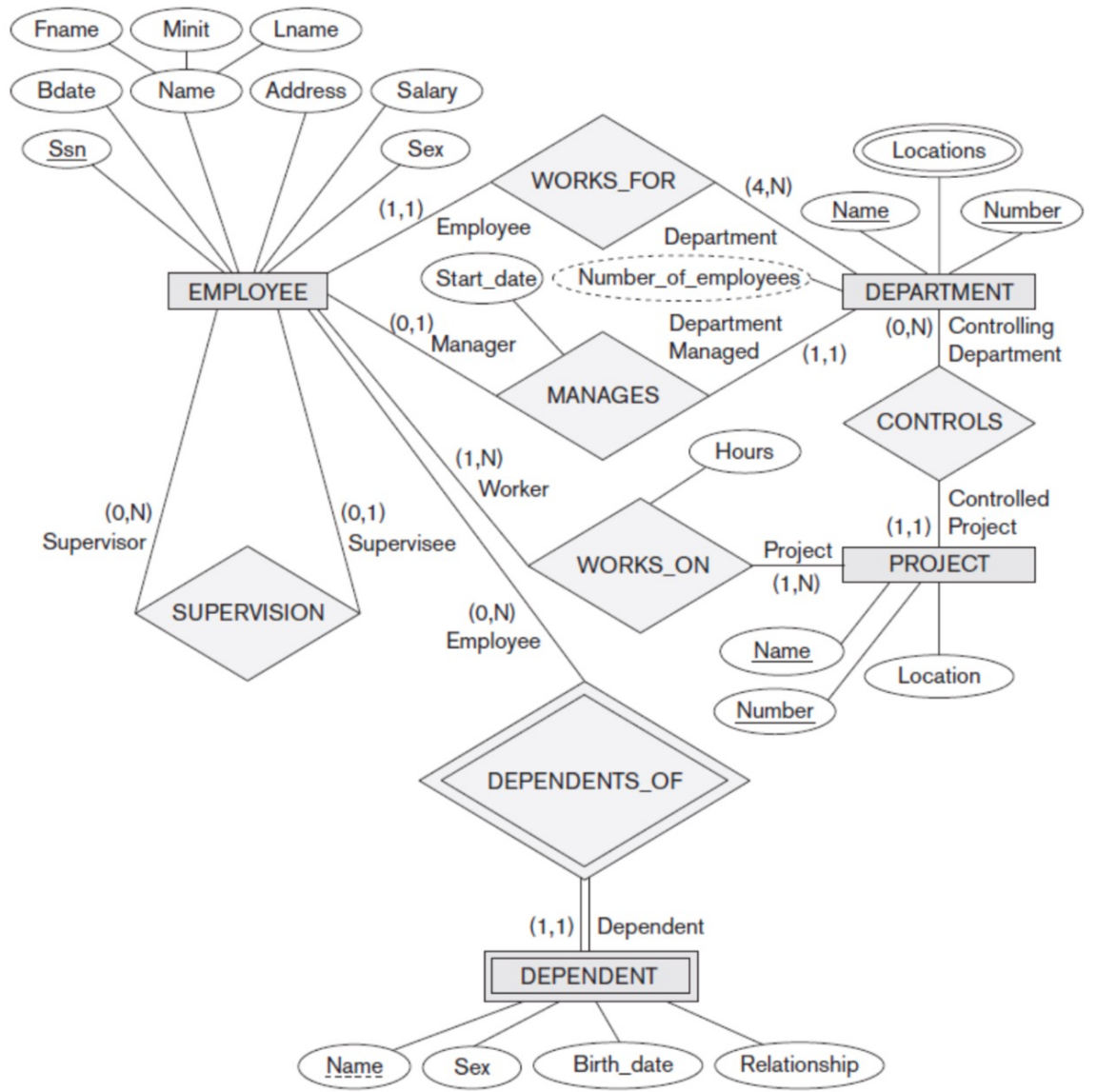


Many-to-Many relationship *enroll*

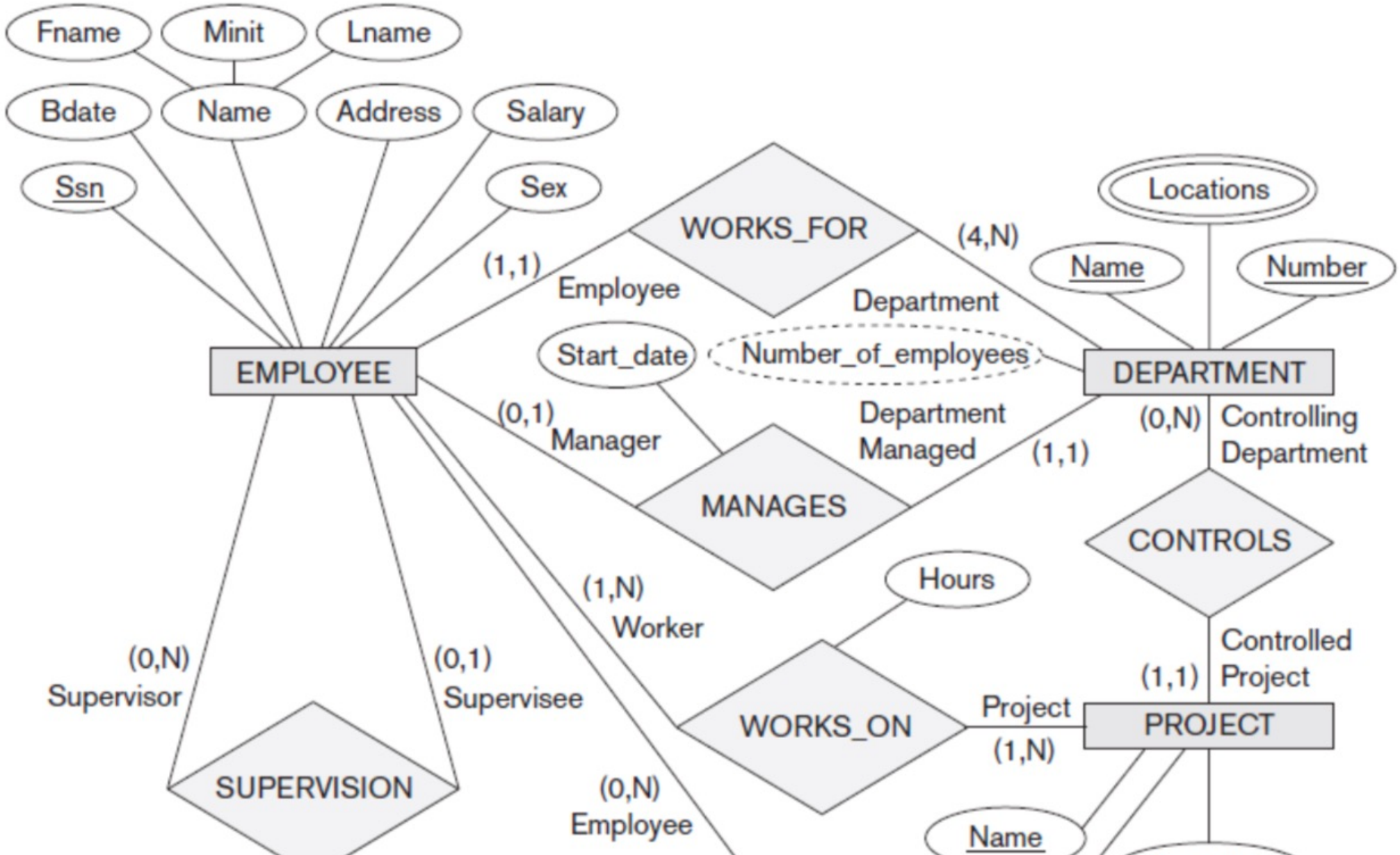
- Already have tables STUDENT and CLASS
- Create a new table ENROLL
  - with primary keys of STUDENT and CLASS (as foreign keys) forming the primary key of ENROLL
  - Relationship attributes (ENROLL_DATE and ENROLL_GRADE) are added as attributes to the table
- Note that MySQL Workbench does so automatically when a relationship is created

Example

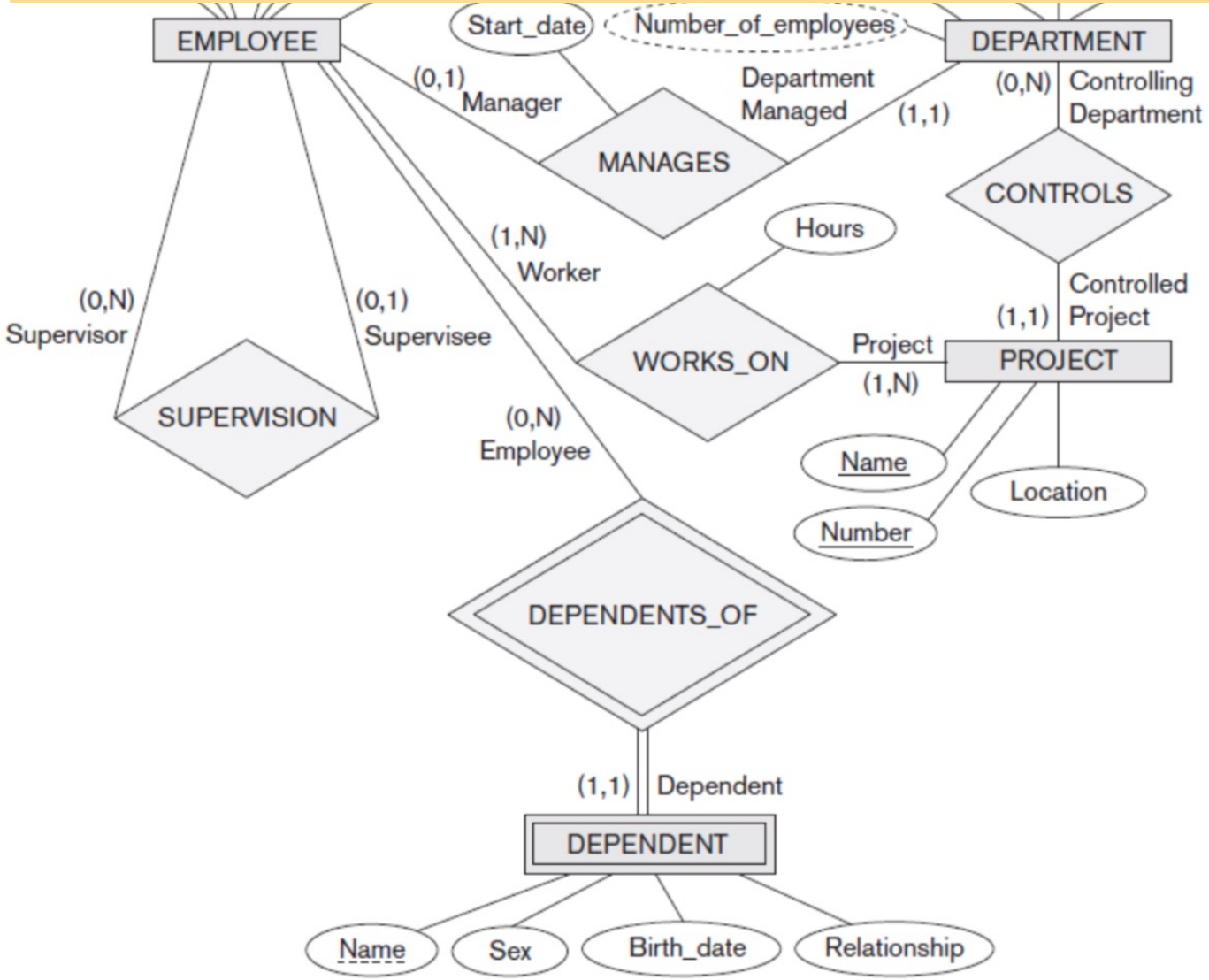
Note the notation for 1-1, 1-M, and M-N relationships in terms of minimum and maximum cardinalities



Example

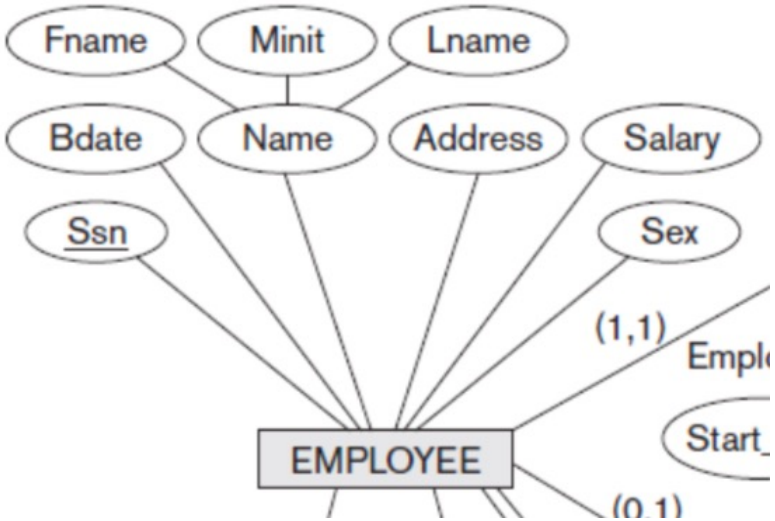

Fname
Minit
Lname
Name
Bdate
Address
Salary
Ssn
Sex
WORKS_FOR
(1,1)
Employee
(4,N)
Department
Locations
Name
Number
Start_date
Number_of_employees
EMPLOYEE
DEPARTMENT
(0,1)
Manager
Department
Managed
(1,1)
(0,N)
Controlling
Department
MANAGES
CONTROLS
Hours
(1,N)
Worker
Controlled
Project
(1,1)
(0,N)
Supervisor
(0,1)
Supervisee
WORKS_ON
Project
(1,N)
PROJECT
SUPERVISION
(0,N)
Employee
Name

Example


EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
(0,1)
Manager
Department
Managed
(1,1)
(0,N)
Controlling
Department
MANAGES
CONTROLS
Hours
(1,N)
Worker
(0,N)
Supervisor
(0,1)
Supervisee
Controlled
Project
(1,1)
WORKS_ON
Project
(1,N)
PROJECT
SUPERVISION
(0,N)
Employee
Name
Location
Number
DEPENDENTS_OF
(1,1)
Dependent
DEPENDENT
Name
Sex
Birth_date
Relationship

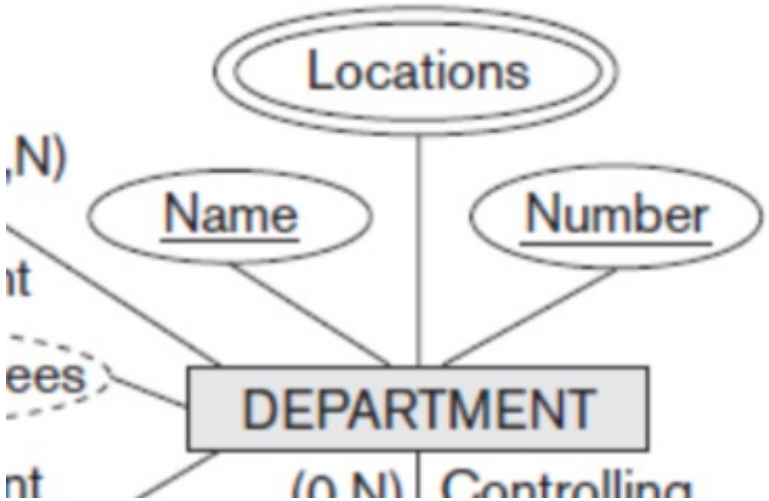
Example … continued – Composite Attribute

Entity EMPLOYEE … table EMPLOYEE



- Attributes
  - Ssn (chosen as primary key)
    - under the assumption that there is no regulation to prevent the use of Ssn to identify an employee
    - Ssn … Social Security Number in USA
  - Bdate
  - Fname, Minit, Lname
    - As parts of composite attributes
  - Address
  - Salary
  - Sex
    - Should be called Gender

Example … continued – Multi-valued Attribute



Entity DEPARTMENT … table DEPARTMENT

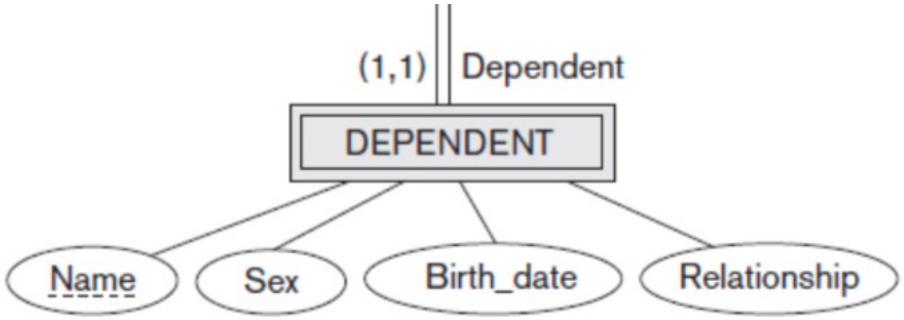
- Attributes
  - *Number* (chosen as primary key)
  - *Name* (unique)
  - Note that the derived attribute Number_of_employees is not an actual attribute stored in the DB … it is calculated/derived when needed

New table DEPT_LOCATIONS due to multi-valued attribute

Locations

- Attribute *Number* from table DEPAERTMENT (FK, PK)
- Attribute *Location* (PK)

Example … continued – Weak Entity

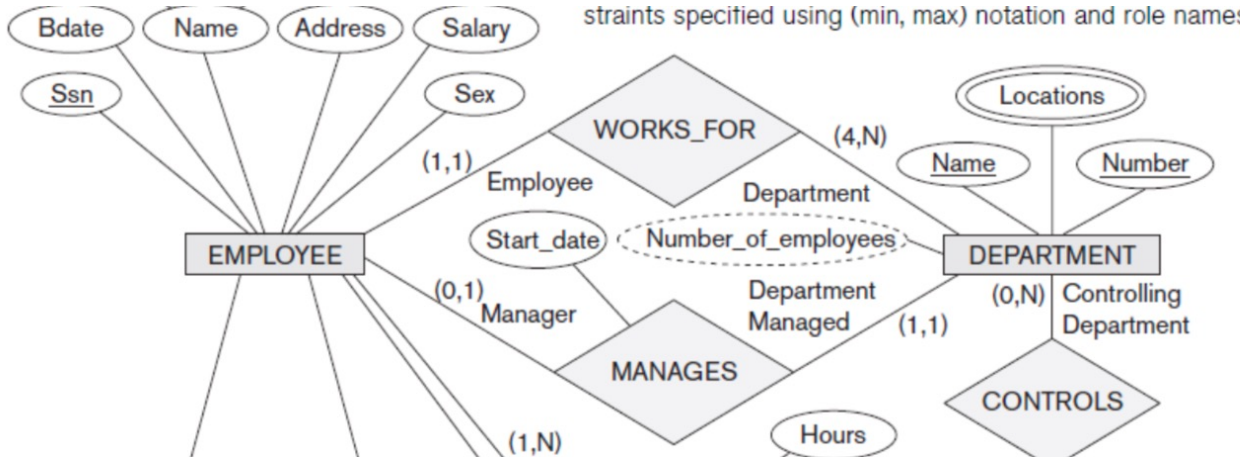


Weak Entity DEPNENDANT

- As the entity has an attribute that has unique values for instances that depend on the same strong entity (names of dependants of one employee are unique), attributes of the weak entity are attributes of its corresponding table

Example … continued -- Relationships

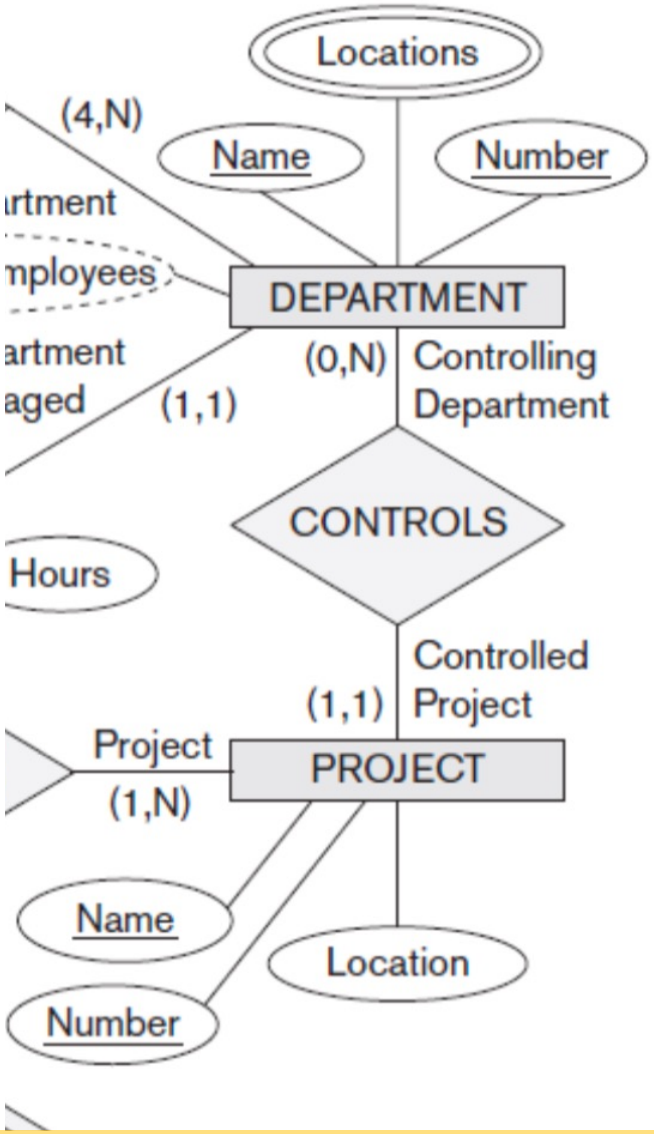
straints specified using (min, max) notation and role names.



- 1-M relationship WORKS_FOR between EMPLOYEE and DEPARTMENT
  - Dept has minimum 4 or many employees
  - Employee works for exactly one department
  - Add Number (primary key of DEPARTMENT (from ”one side”) to table EMPLOYEE (many-side of the relationship)

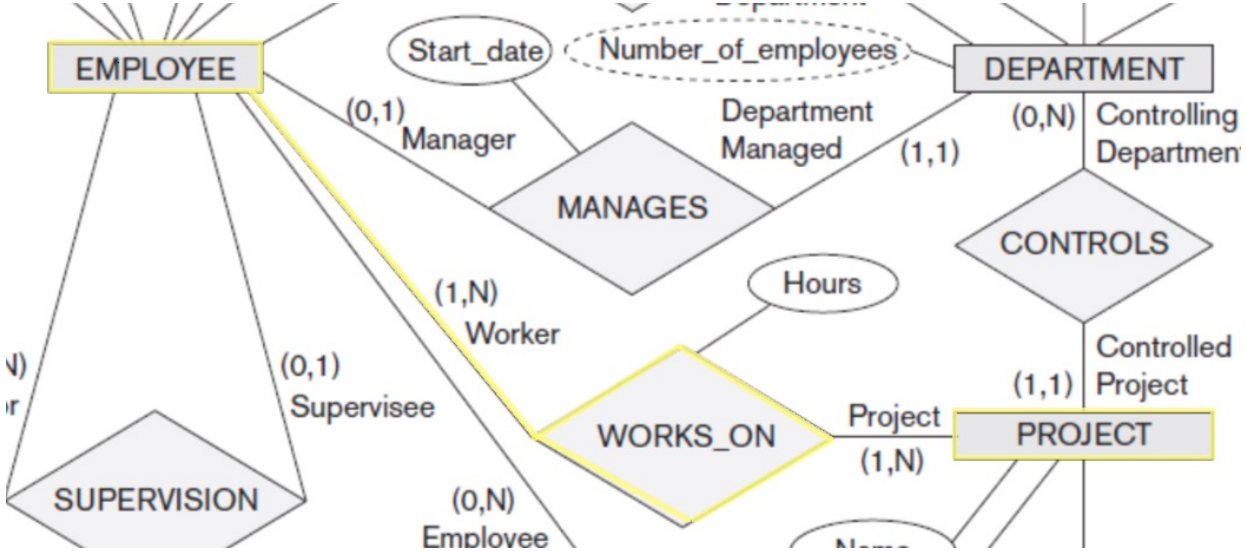
- 1-1 relationship MANAGES between EMPLOYEE and DEPARTMENT
  - Dept is managed by exactly one employee
  - Employee manages at most one department
  - Options:
    1. Add primary key of EMPLOYEE table to DEPARTMENT table
    2. Add primary key of DEPARTMENT table to EMPLOYEE table
  - Choose option 1 as it results in fewer NULL values

Example … continued -- Relationships



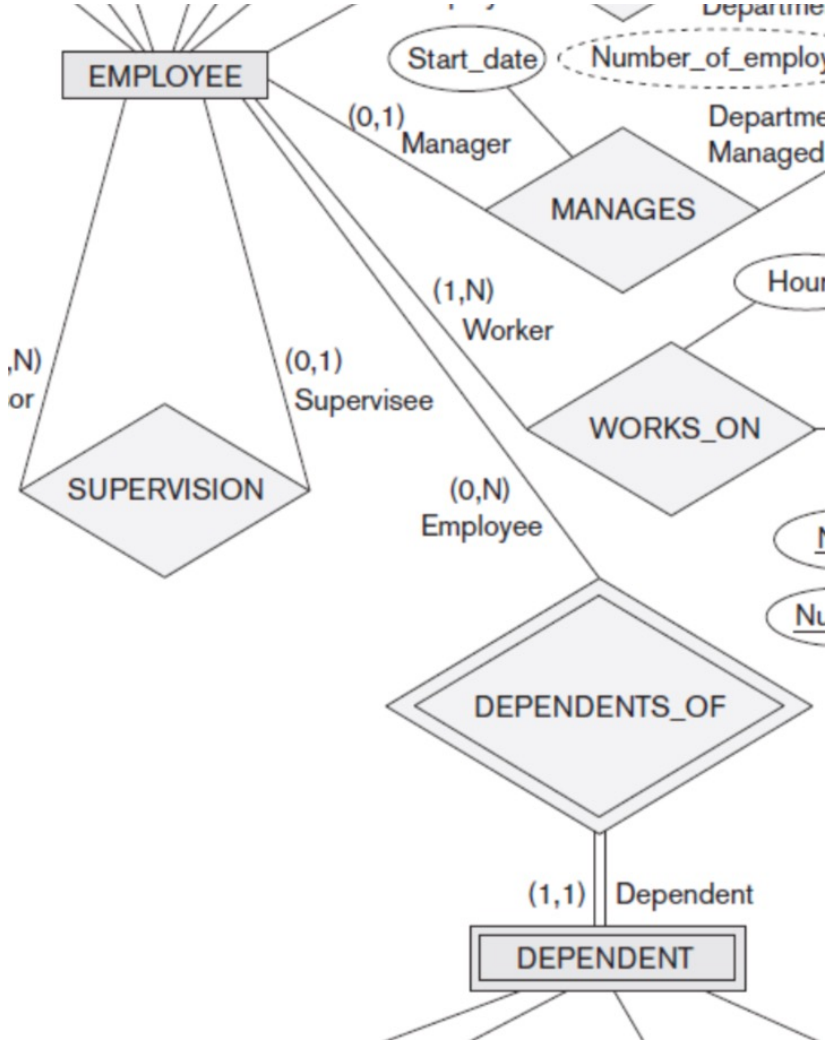
- 1-M relationship CONTROLS between DEPARTMENT and PROJECT
  - Dept controls zero, one, or more projects
  - Project is controlled by exactly one department
- Add NUMBER (primary key of DEPARTMENT from ”one side”) to table PROJECT (many-side of the relationship) … but call it DEPT_NUMBER

Example … continued -- Relationships



- N-M (many-to-many) relationship WORKS_ON between EMPLOYEE and PROJECT with attribute *Hours*
  - Employee works on one or more project
  - Project is worked on one or more employees
- Create a table WORKS_ON containing
  - Primary key *Number* from table PROJECT
  - Primary key *Ssn* from table EMPLOYEE
    - Each of *Number* and *Ssn* is a foreign key and together they form a composite primary key
  - Attribute/column *Hours*

Example … continued -- Relationships



- 1-N (one-to-many) relationship DEPENDENTS_OF between EMPLOYEE and DEPENDENT (weak entity)
  - Employee has zero, one, or more depndants
  - Dependant depends exactly on one employee
- Add Ssn (primary key of EMPLOYEE (from ”one side”)) to table DEPENDENT (many-side of the relationship)

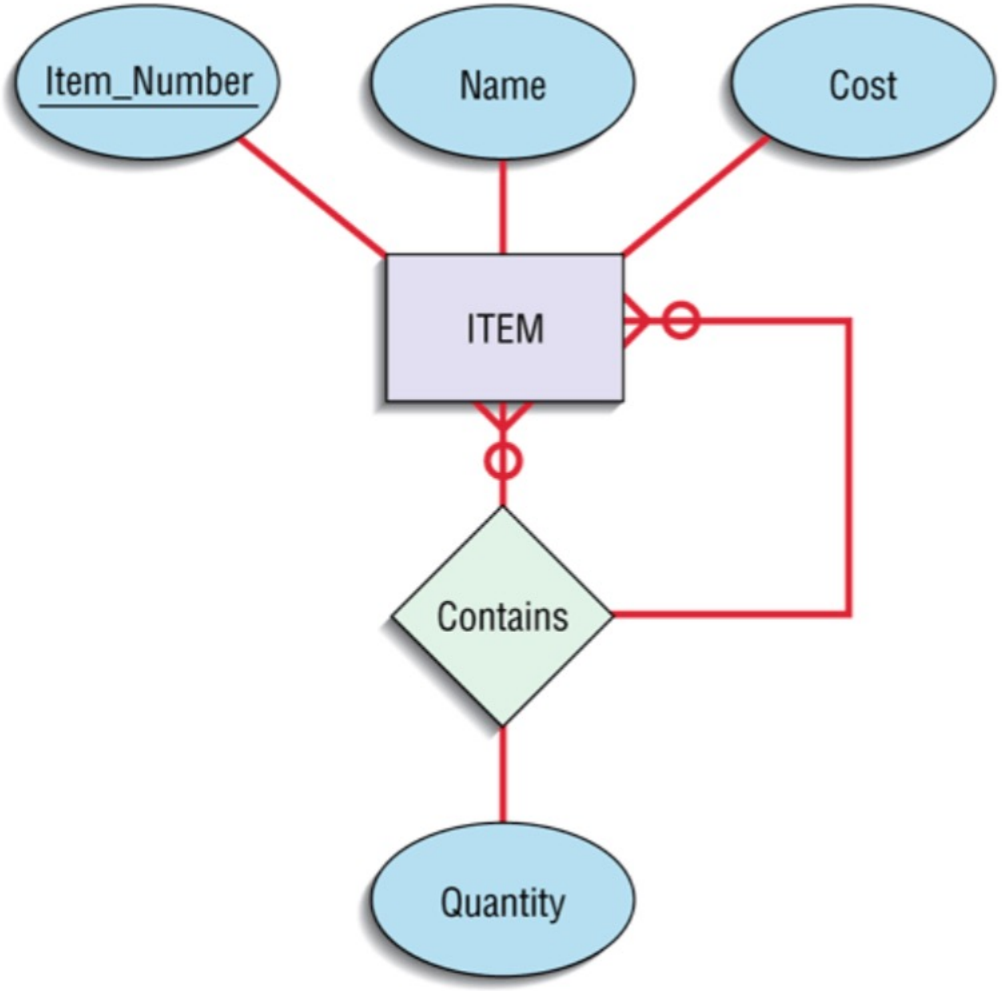
Example … continued

- Document Business Rules
  - State which ones will be enforced automatically by the DBMS once the tables are created … this will be done later, after we cover SQL
- Business rules
  - All constraints derived from the ER diagram

Example – Unary Relationships

- M-N (many-to-many) Unary Relationship
  CONTAINS between ITEM and ITEM
  - Item may contain 0, 1, or more items
  - Item may be contained in 0, 1, or more items

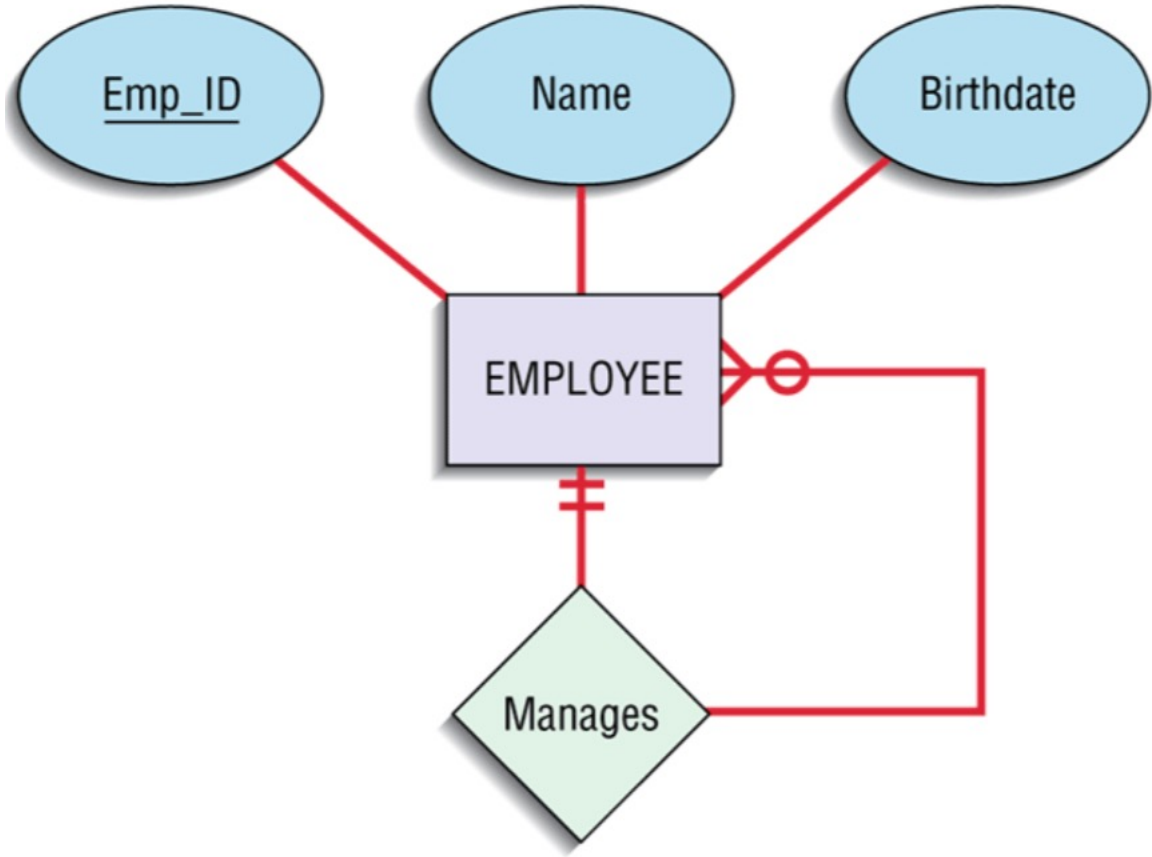
- Create a table CONTAINS with attributes
  - primary key Item_Number
  - Primary key of ITEM … call it Contained_Item_Number



Example – Unary Relationships … continued

- 1-N (one-to-many) Unary/Recursive Relationship *Manages* between EMPLOYEE and EMPLOYEE
  - Employee may manage 0, 1, or more employees
  - Employee is managed by exactly one employee

- Add a primary key from "one-side" to "many-side"
  - Add Emp_ID to the table EMPLOYEE –call it Managed_by_Emp



Questions and Answers (Q/A)