Chapter 7: Entity-Relationship Model (contd)

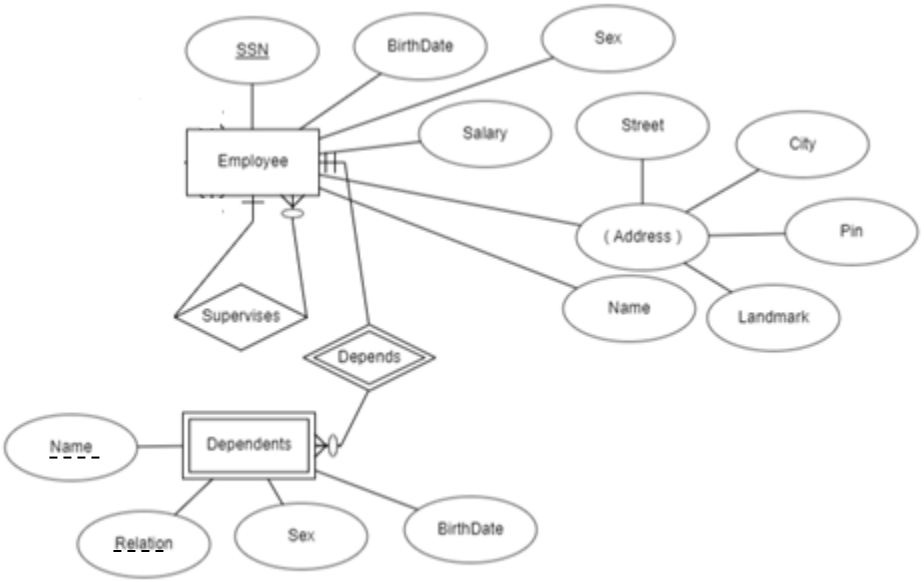
Database System Concepts, 6th Ed.

Weak Entity Sets

- An entity set that does not have a primary key is referred to as a **weak entity set**.
- The existence of a weak entity set depends on the existence of a **identifying entity set**
  - It must relate to the identifying entity set via a total, one-to-many relationship set from the identifying to the weak entity set
  - **Identifying relationship** depicted using a double diamond
- The **discriminator** (*or partial key)* of a weak entity set is the set of attributes that distinguishes among all the entities of a weak entity set.
- The primary key of a weak entity set is formed by the primary key of the strong entity set on which the weak entity set is existence dependent, plus the weak entity set's discriminator.

Weak Entity Sets (Cont.)

- We underline the discriminator of a weak entity set with a dashed line.
- We put the identifying relationship of a weak entity in a double diamond.
- Primary key for *section* – (*ssn, name, relation*)



Weak Entity Sets (Cont.)

- Note: the primary key of the strong entity set is not explicitly stored with the weak entity set, since it is implicit in the identifying relationship.
- If *ssn* were explicitly stored, *dependent* could be made a strong entity, but then the relationship between *dependent* and *employee* would be duplicated by an implicit relationship defined by the attribute *ssn* common to *dependent* and *employee*

Redundant Attributes

- Suppose we have entity sets
  - *employee*, with attributes including *dnumber*
  - *department*

  and a relationship
  - *emp_dept* relating *employee* and *department*
- Attribute *dnumber* in entity *instructor* is redundant since there is an explicit relationship *emp_dept* which relates instructors to departments
  - The attribute replicates information present in the relationship, and should be removed from *employee*
  - BUT: when converting back to tables, in some cases the attribute gets reintroduced, as we will see.

Tutorial

Construct an E-R diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents. Each insurance policy covers one or more cars, and has one or more premium payments associated with it. Each payment is for a particular period of time, and has an associated due date, and the date when the payment was received.

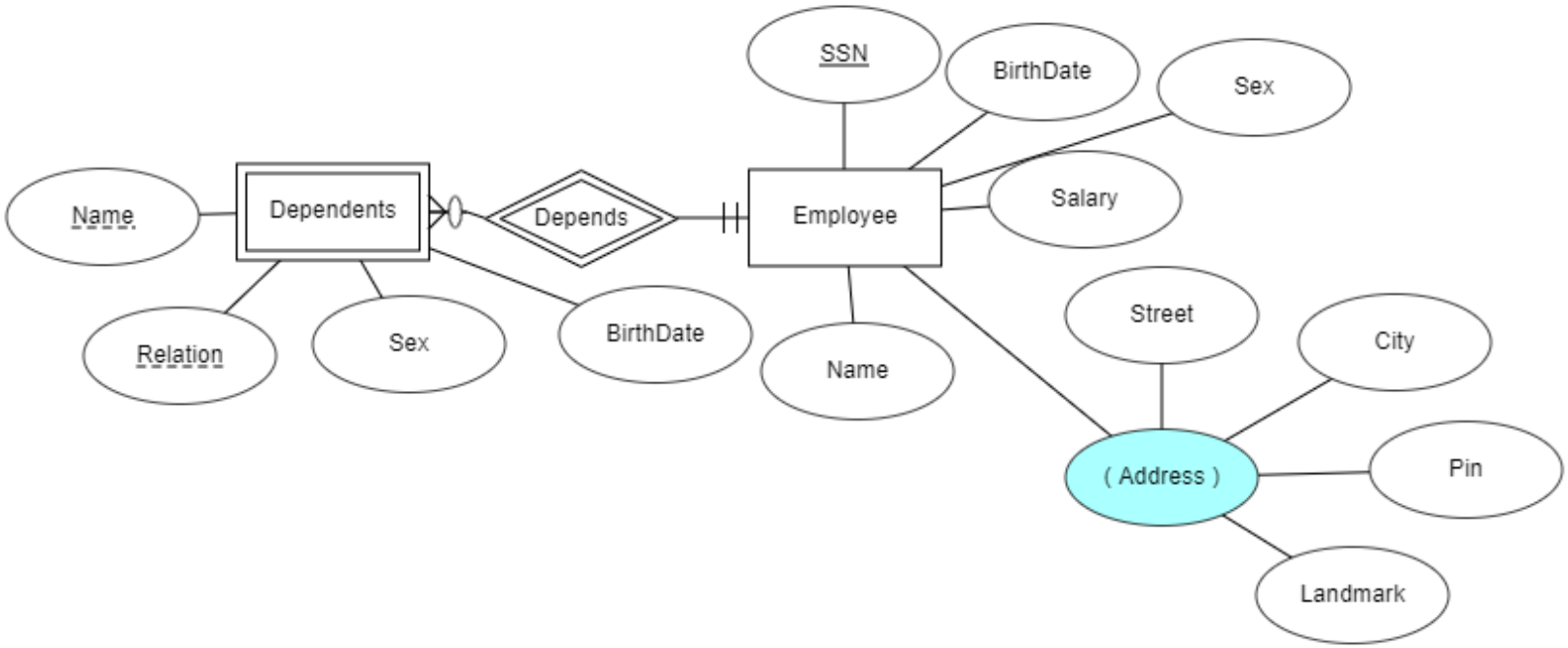
Reduction to Relational Schemas

Reduction to Relation Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collection of schemas.
- For each entity set and relationship set there is a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

Representing Entity Sets With Simple Attributes

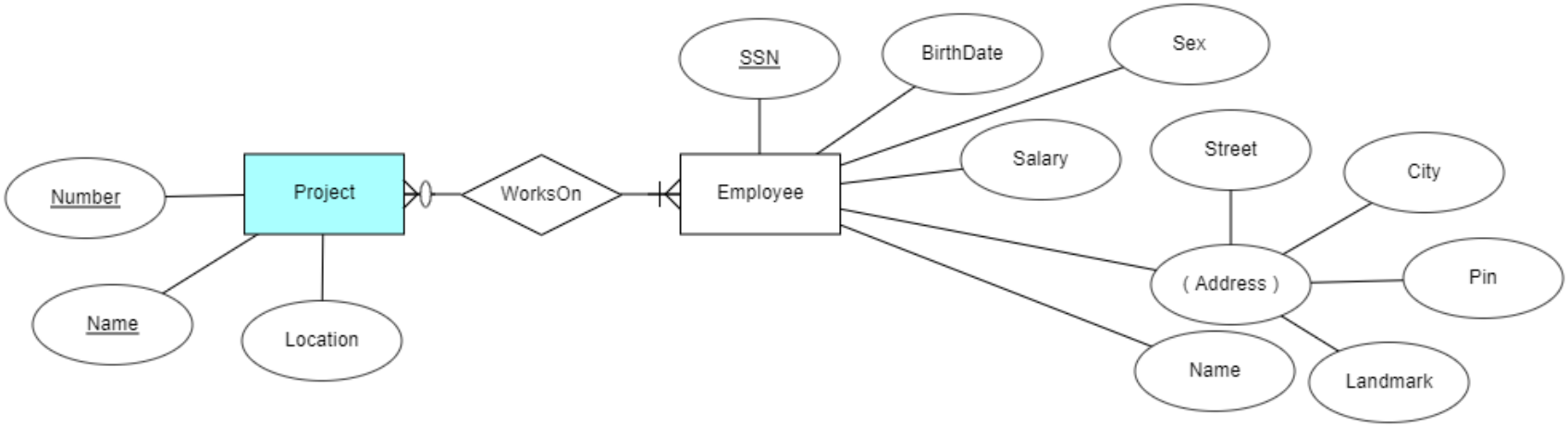
- A strong entity set reduces to a schema with the same attributes
  Employee(ssn, name, sex, birthdate, salary, street, city, landmark, pin)
- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set
  Dependent(ssn,name, relation, sex, birthdate)



Representing Relationship Sets

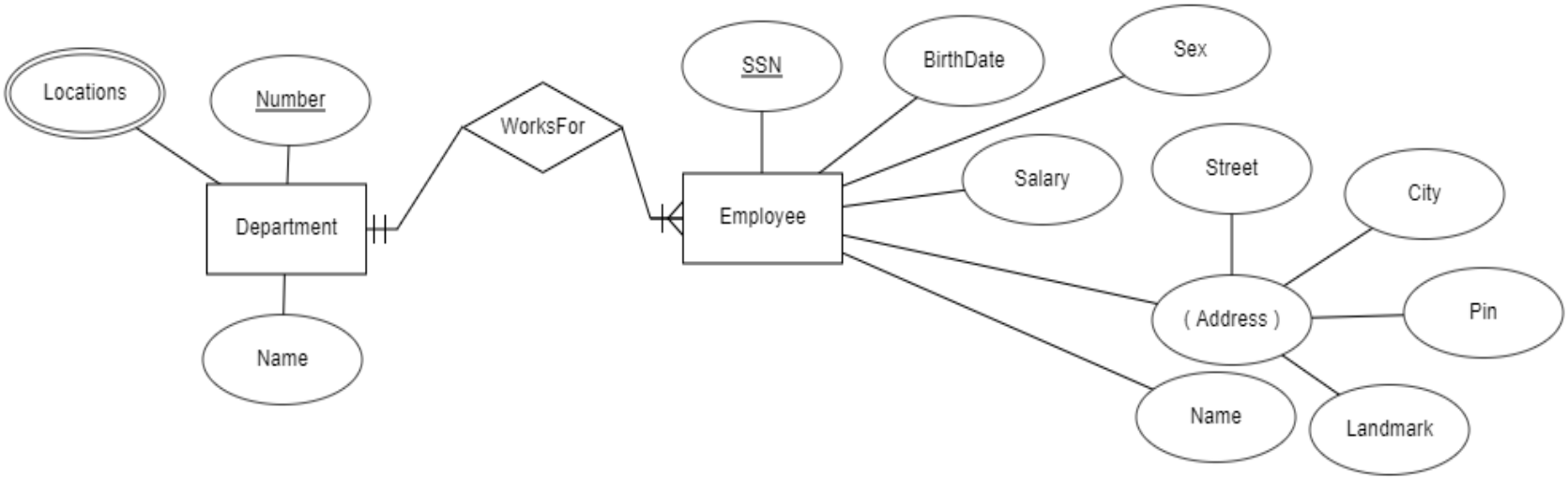
- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set *worksOn*

  WorksOn(<u>ssn, pnumber, pname</u>)



Redundancy of Schemas

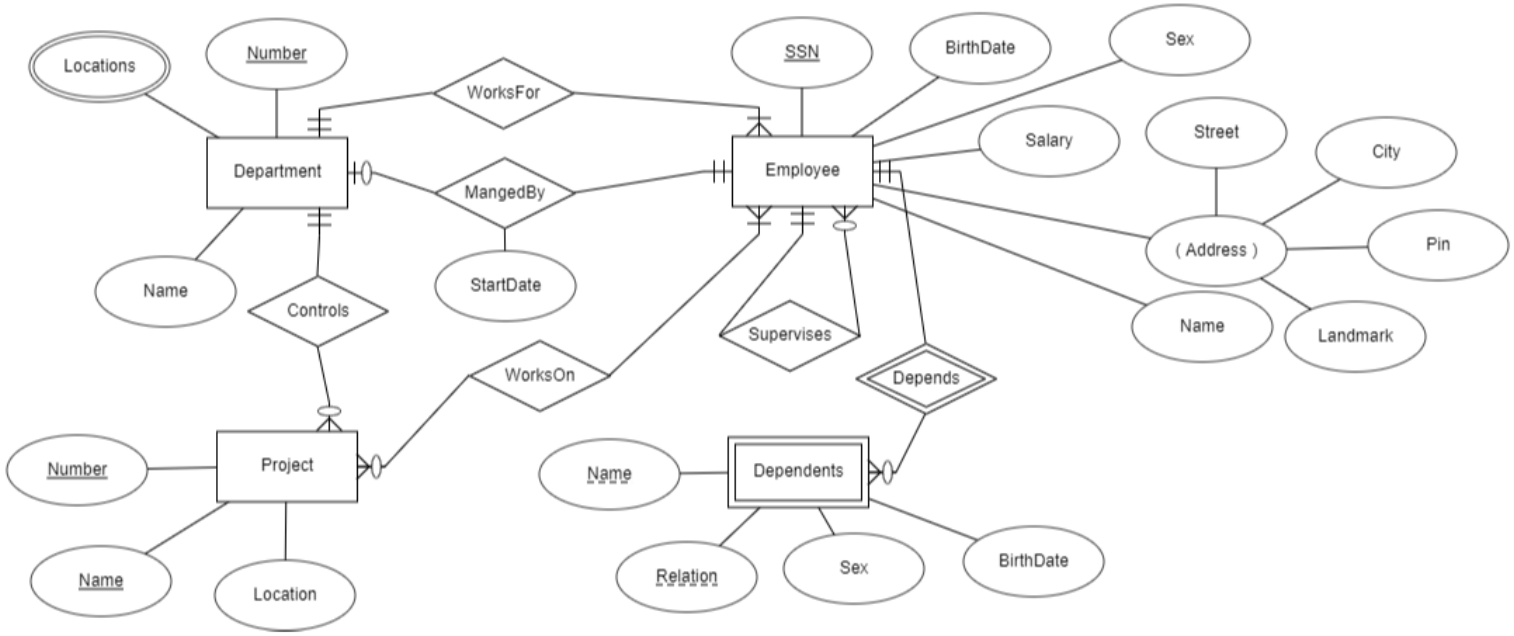
- Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side
- Example: Instead of creating a schema for relationship set *emp_dept*, add an attribute *dnumber* to the schema arising from entity set *employee*



Redundancy of Schemas (Cont.)

- For one-to-one relationship sets, either side can be chosen to act as the “many” side
  - That is, extra attribute can be added to either of the tables corresponding to the two entity sets
- If participation is *partial* on the “many” side, replacing a schema by an extra attribute in the schema corresponding to the “many” side could result in null values
- The schema corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.
  - Example: The *dependent* schema already contains the attributes that would appear in the *dependent_employee* schema

Reduction to Relation Schemas



Employee(<u>ssn</u>, name, sex, birthdate, salary, street, city, landmark, pin, dnumber, supervisorSsn)

Project (<u>number, name</u>, location, dnumber)

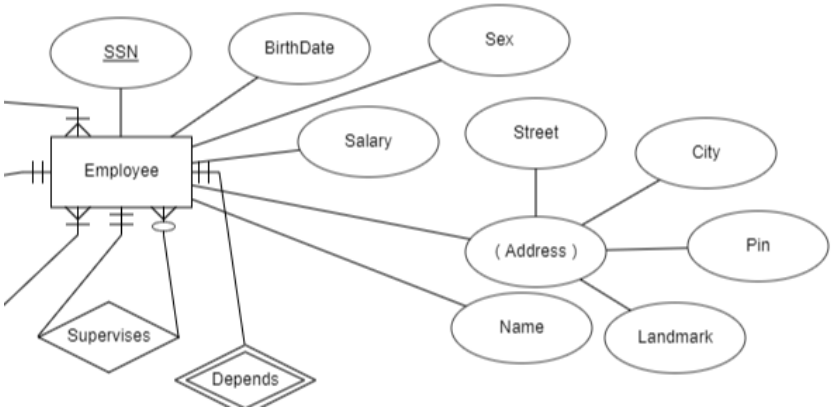
Department(<u>number</u>, name, mgrSsn, mgrStartdate)

Dependent(<u>ssn,name, relation</u>, sex, birthdate)

DeptLocations(<u>number, location</u>)

WorksOn(<u>ssn, pnumber, pname</u>)

Composite and Multivalued Attributes

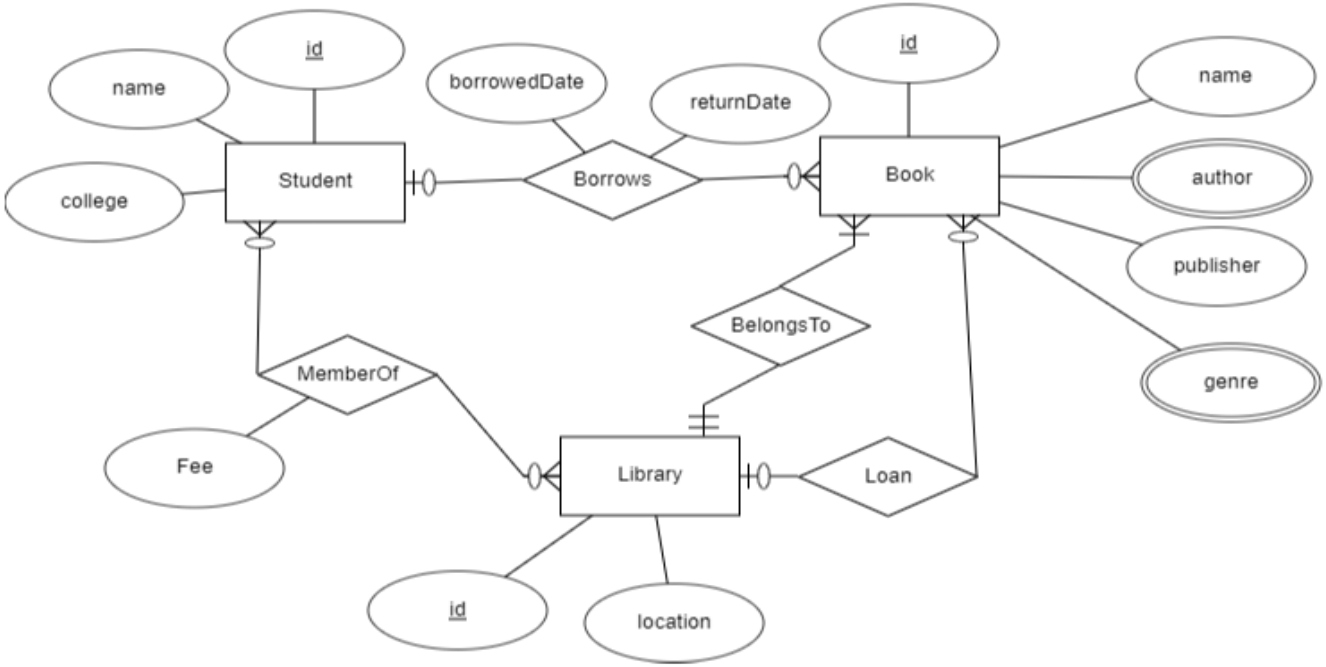


- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - Example: given entity set *employee* with composite attribute *address* with component attributes street, *city*, *pin* and *landmark* the schema corresponding to the entity set has four attributes street, *city*, *pin* and *landmark*
    - *Prefix omitted if there is no ambiguity*
- Ignoring multivalued attributes, extended employee schema is
- **Employee**(ssn, name, sex, birthdate, salary, street, city, landmark, pin)

Composite and Multivalued Attributes

- A multivalued attribute *M* of an entity *E* is represented by a separate schema *EM*
  - Schema *EM* has attributes corresponding to the primary key of *E* and an attribute corresponding to multivalued attribute *M*
  - Example: Multivalued attribute locations of *departments* is represented by a schema:
    DeptLocations(number, location)
  - Each value of the multivalued attribute maps to a separate tuple of the relation on schema *EM*
    - For example, a *department* entity with primary key CSE and locatins AB5 and AB2 maps to two tuples:
      (CSE, AB5) and (CSE, AB2)

Tutorial



Student(id, name, college)
Book(id, name, publisher, sid, borrow_date, return_date, lid, loaned_to)
Library(id, location)
MemberOf(sid, lid, fee)
Author(bid, author_name)
Genre(bid, genre_name)