Logical Database Design (ER to Relational Model mapping)

Learning Outcomes

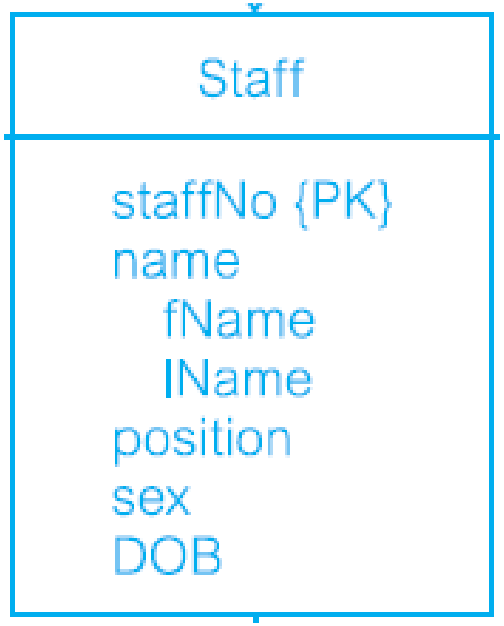
- Understand how to derive a set of relations from a conceptual data model (ER model)
- Able to map an ER model to a set of relations (or relational schema)

Mapping ER model concepts to relations

- Entity
- Binary 1:1, 1:*, *:* relationships
- Complex relationships
- Multi-valued attributes
- Superclass/subclass relationships

Entity

- For each entity:
  - create a relation that includes all the attributes of that entity.
  - For composite attributes, include only the constituent simple attributes (broken into several columns).

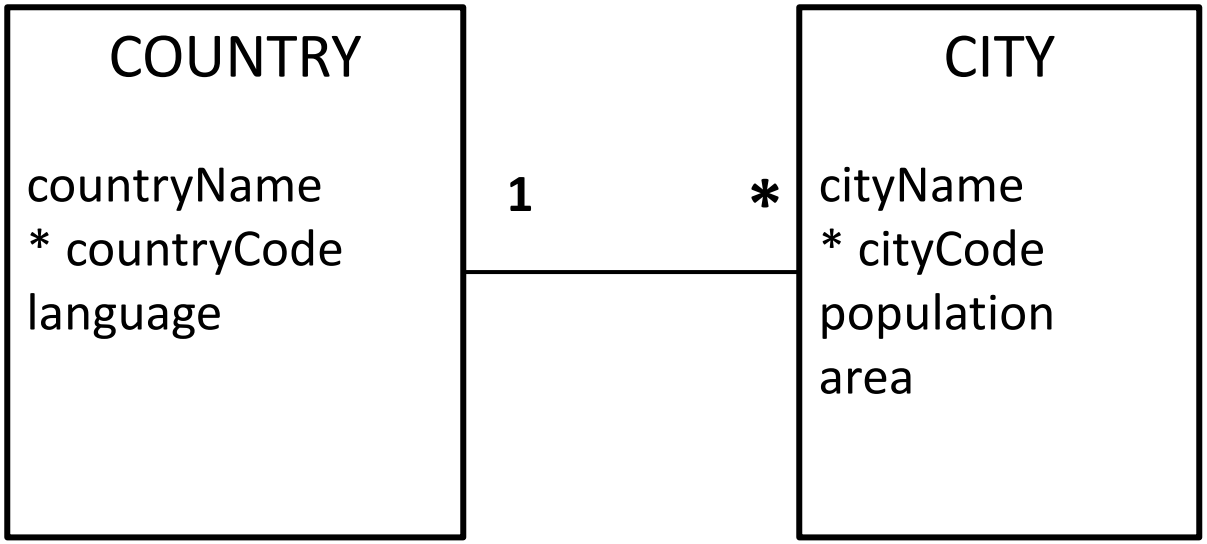


Relational schema of Staff Entity:

Staff (staffNo, fName, lName, position, sex, DoB)

One-to-many (1:*) binary relationships

- Post a copy of the primary key attribute(s) of one-side entity into the relation representing the many-side, to act as a foreign key.

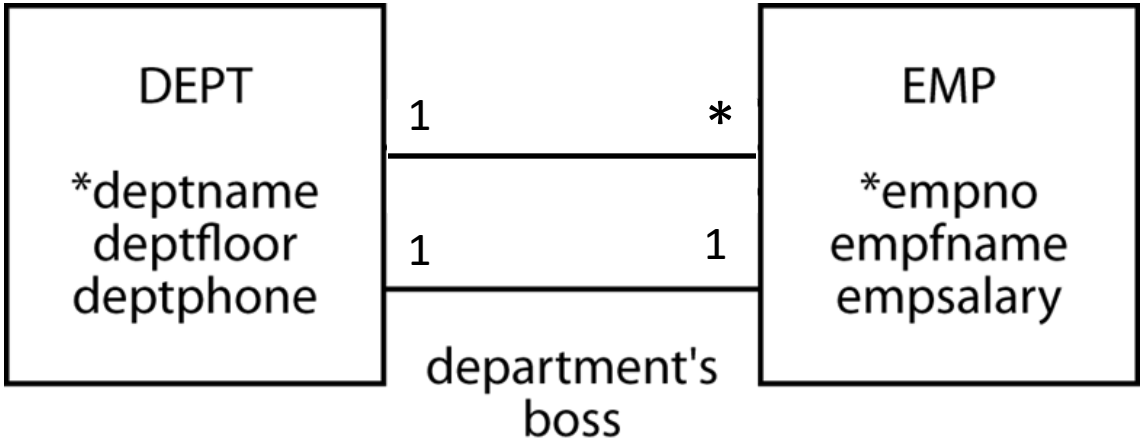


Relational schema for the two entities and 1:* relationship:

Country (<u>countryCode</u>, countryName, language)

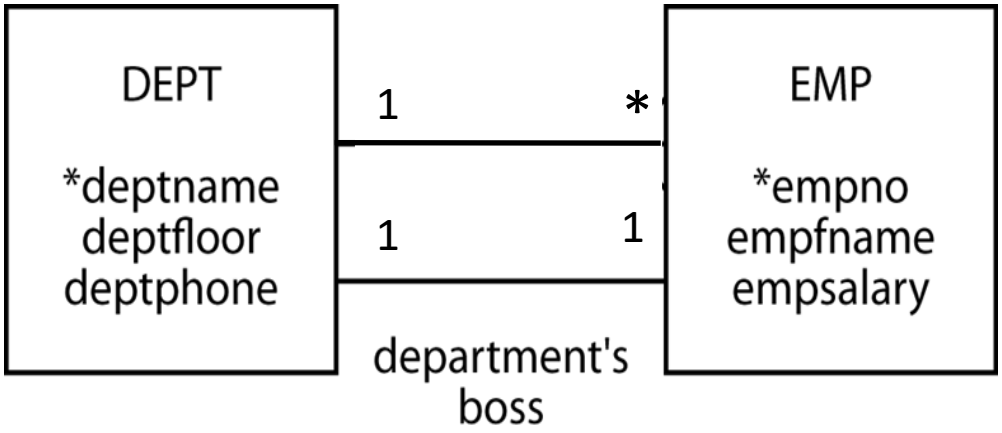
City (<u>cityCode</u>, cityName, population, area, *countryCode*)

One-to-one (1:1) binary relationships



- Mapping 1:1 relationship follows the same rule:
  - Place foreign keys in one or both relations
- Three alternatives for this example.

One-to-one (1:1) binary relationships



- Put the foreign key in Dept.

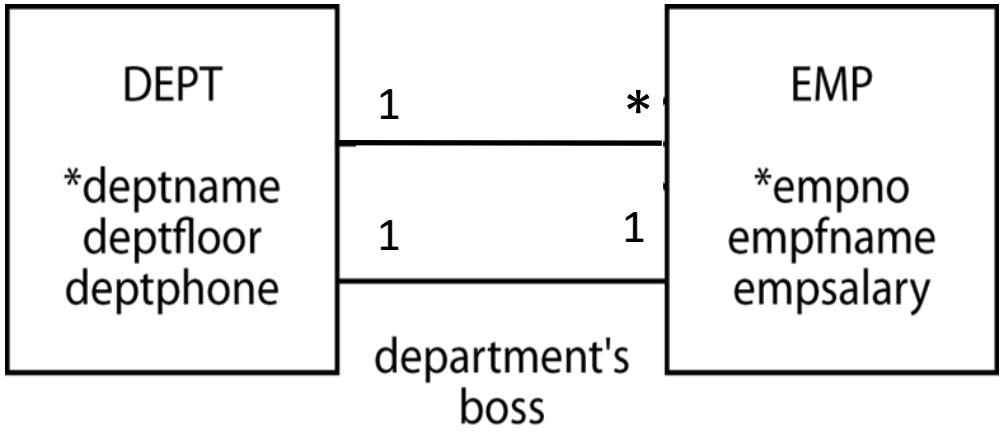
  Every instance of *Dept* will record *empno* of the employee who is the boss. All depts have boss so the foreign key will not be null.

- Put the foreign key in Emp.

  Every instance of *Emp* should record *deptname* of the department this employee manages. Most employees are not bosses so this is usually null.

- Put a foreign key in both Dept and Emp.

One-to-one (1:1) binary relationship



- Sound approach: select entity that results in fewest nulls (less confusing to clients).
- This example: simplest approach is to put the foreign key in Dept.

Dept(deptname, deptfloor, deptphone, *empno*)

Emp(empno, empfname, empsalary, *deptname*)

empno in Dept is the boss empno for a department, it is the foreign key indicating the 1:1 relationship.

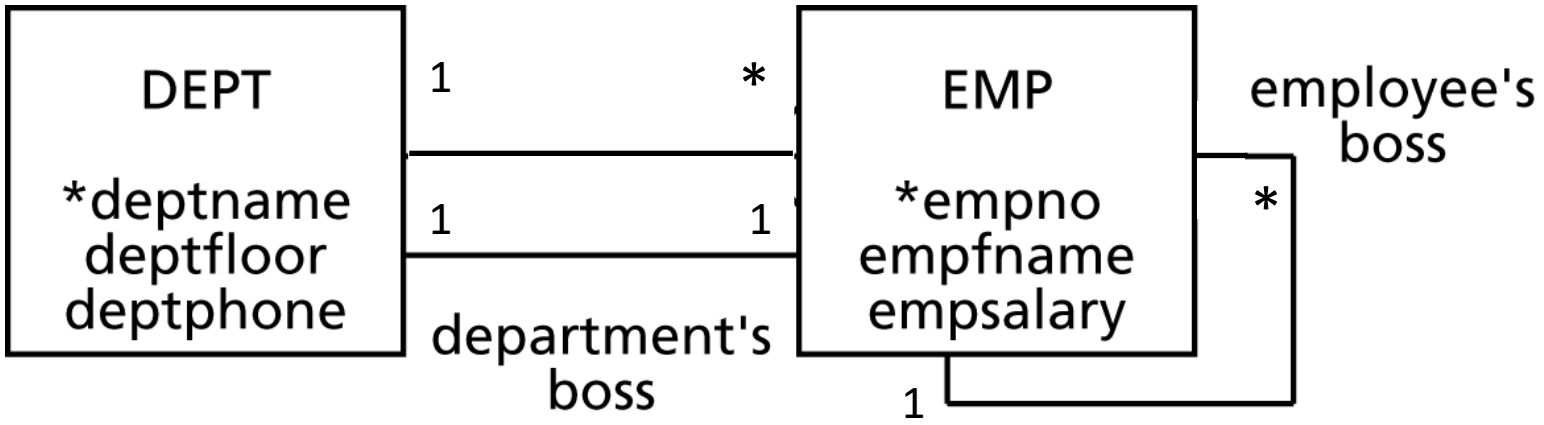
Deptname in Emp is the department a employee works at, it is the foreign key indicating the 1: relationship.*

One-to-one (1:1) relationships mapping rules

- ***mandatory participation* on both sides**,
  - Combine the entities involved into one relation
  - Choose one primary key of the original entities as the primary key of the new relation, the other primary key is candidate key.
- ***mandatory participation* on only one side**,
  - Two relations: each entity is a relation
  - Copy of the primary key from optional participation relation is placed in the mandatory participation relation, as foreign key
- ***optional participation* on both sides**,
  - new relation to represent the relationship.
  - Two attributes, copies of primary key from both relations. Copies of the primary keys act as foreign keys and have to be renamed to indicate the purpose of each

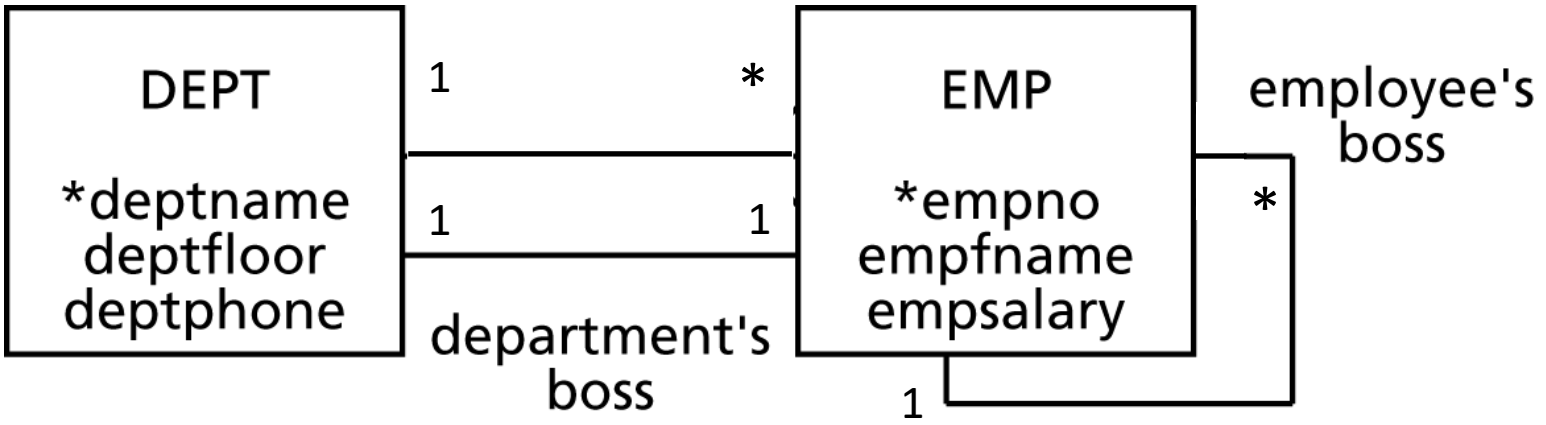
One-to-many (1:*) recursive relationships

- For a 1:* recursive relationship, follow same rules for a 1:* relationship. Add column(s) for foreign key, for the entity at the “many ” side of the relationship.



- What’s the new *emp* relation?

Test your understanding



Previously without the 1:* recursive relationship, we have
Dept(deptname, deptfloor, deptphone, *empno*)
Emp(empno, empfname, empsalary, *deptname*)

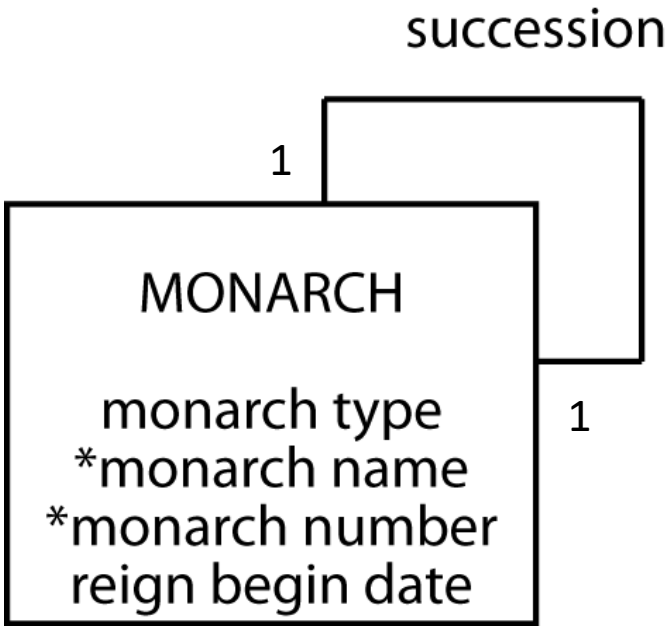
- What's the new *emp* relation?

One-to-one (1:1) recursive relationships

- ***mandatory participation* on both sides**,
  - Represent recursive relationship as one relation with two copies of primary key, one primary key serves as a foreign key and should be renamed to indicate the recursive relationship.
- ***optional participation* on both sides**,
  - new relation to represent the relationship. Two attributes, both copies of primary key. Copies of the primary keys act as foreign keys and have to be renamed to indicate the purpose of each
- ***mandatory participation* on only one side**,
  - a) A single relation with two copies of the primary key (as when both mandatory).
  - Or b) create a new relation to represent the relationship (as when both optional).

One-to-one (1:1) recursive relationships

- The English monarch



The successor (person to come after) for a monarch is also a monarch

One-to-one (1:1) recursive relationships

- The monarch relation

Primary key
(name and number)

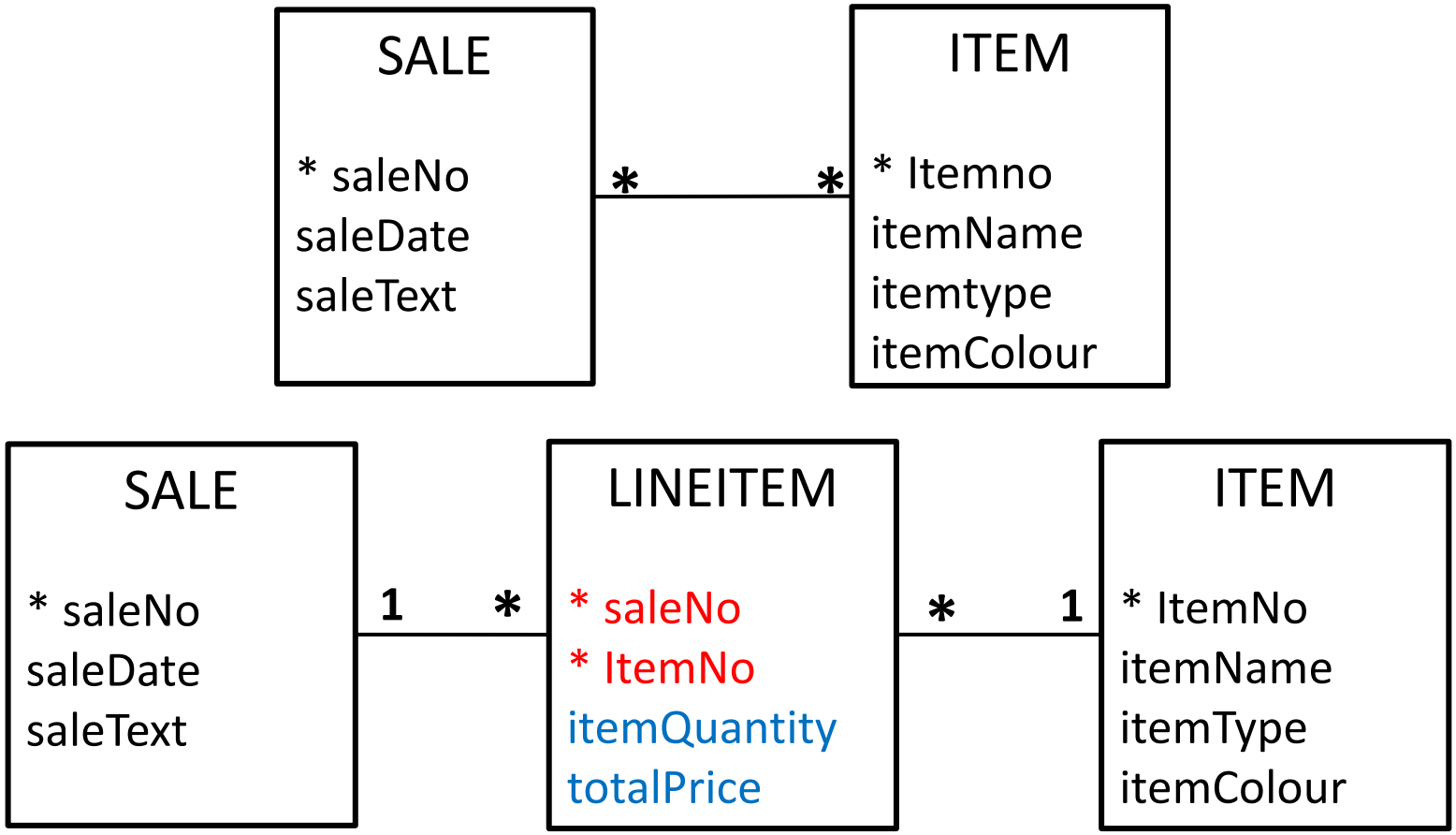
Predecessor
(number and name)
This is foreign key in same table

monarch					
montype	monname	monnum	rgnbeg	*premonname*	*premonnum*
Queen	Victoria	I	1837/6/20	William	IV
King	Edward	VII	1901/1/22	Victoria	I
King	George	V	1910/5/6	Edward	VII
King	Edward	VIII	1936/1/20	George	V
King	George	VI	1936/12/11	Edward	VIII
Queen	Elizabeth	II	1952/2/6	George	VI

Many-to-many (*:*) binary relationships

- Create a new relation to represent the relationship
- Foreign Key columns pointing to participating entities (these foreign keys will also form the primary key of the new relation)
- Further columns for attributes of the relationship.

- We covered this in the ER model discussion: create an associative entity for *:* binary relationships (see next slide)

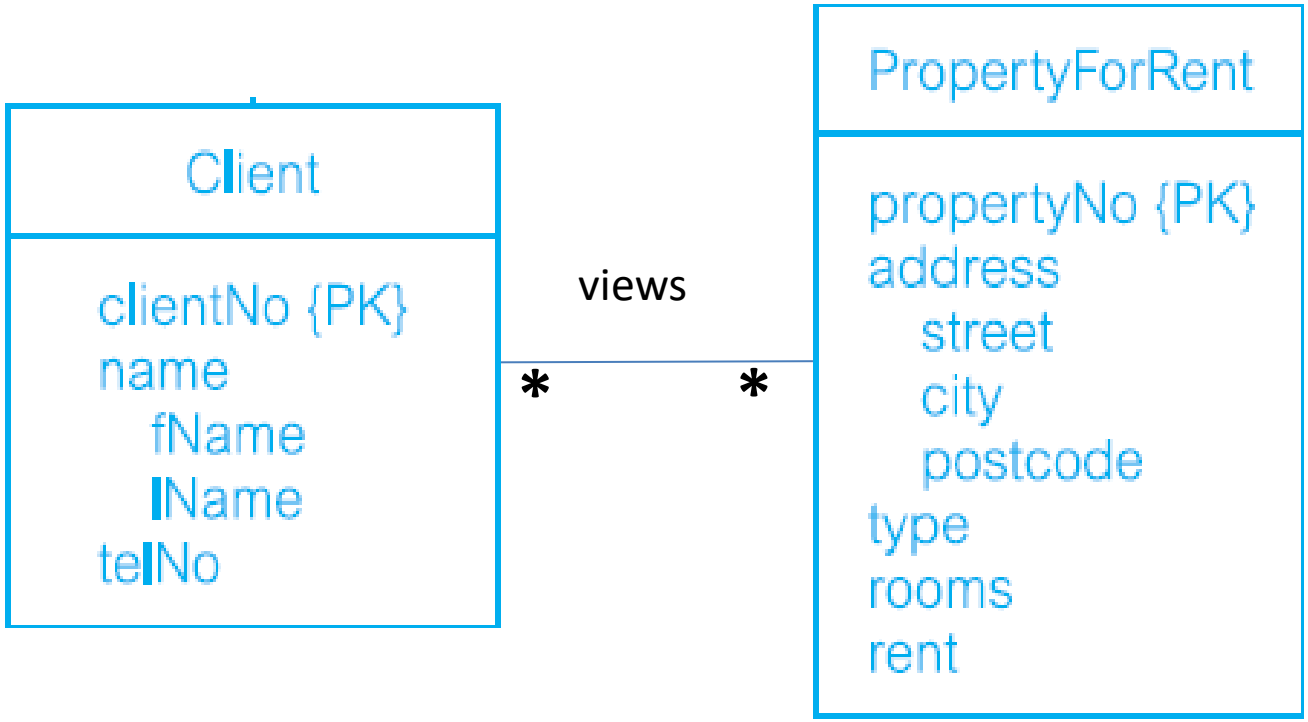


Relational schema of the ER model above:

Sale (saleNo, saleDate, saleText)

Item (ItemNo, itemName, itemType, itemColour)

Lineitem (saleNo, ItemNo, itemQuantity, totalPrice)



- Relational schema for the above ER model:

Client (clientNo, fName, lName, telNo)

PropertyForRent(propertyNo, street, city, postcode, type, rooms, rent)

Viewing(clientNo, propertyNo, dateView, comment)

Complex relationships

- Create a new relation to represent the relationship .
- Include any attributes that are part of the relationship.
- Post a copy of the primary key attribute(s) of the participating entities as foreign keys.
- Any foreign keys that represent a ‘many’ relationship (for example, 1..*, 0..*) usually form primary key of this new relation
- The primary key of the new relation possibly require some of the new attributes

Multi-Valued Attributes

- Create a new relation to represent multi-valued attribute. Include primary key of entity in new relation, to act as a foreign key.
- Example:

  Telephone number in Branch relation:

  Telephone(telNo, BranchNo)

This relation tells us that telNo is a telephone number that belongs in the Branch BranchNo.

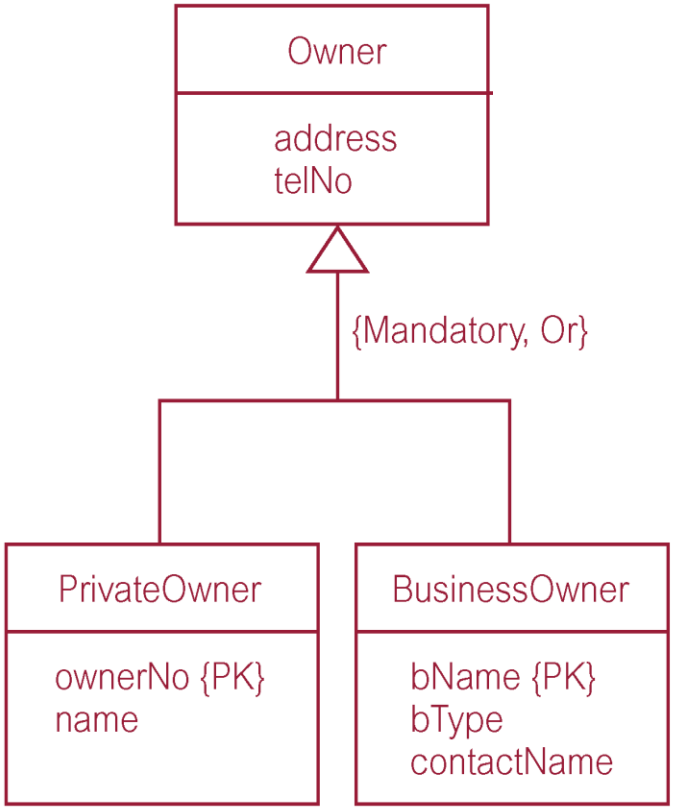
Allows Branch to have many telephone numbers (which could not fit in one attribute).

Superclass/subclass relationships

- Identify superclass entity as the parent entity and subclass entity as the child entity.
  - Various options on how to represent such a relationship as one or more relations.
- Factors to consider:
  - disjointness and participation constraints on the superclass/subclass relationship,
  - whether the subclasses are involved in distinct relationships,
  - number of participants in the superclass/subclass relationship.

Superclass/subclass relationships

Participation constraint	Disjoint constraint	Relations required
Mandatory	Nondisjoint (And)	Single relation with one or more discriminators to distinguish type of each tuple
Optional	Nondisjoint (And)	Two relations: one for superclass, one for all subclasses (one or more discriminators to distinguish type)
Mandatory	Disjoint (Or)	Many relations: one for each combined superclass/subclass
Optional	Disjoint (Or)	Many relations: one relation for superclass and one for each subclass



- Mandatory, or → one relation for each combined superclass/subclass
- So relational schema for the above ER model:

PrivateOwner(<u>ownerNo</u>, name, address, telNo)

BusinessOwner (<u>bName</u>, bType, contactName, address, telNo)

Test your understanding

- How many relations do we need in these subclass, superclass cases?
- A) Mandatory, Nondisjoint (and) 1 superclass 5 subclasses

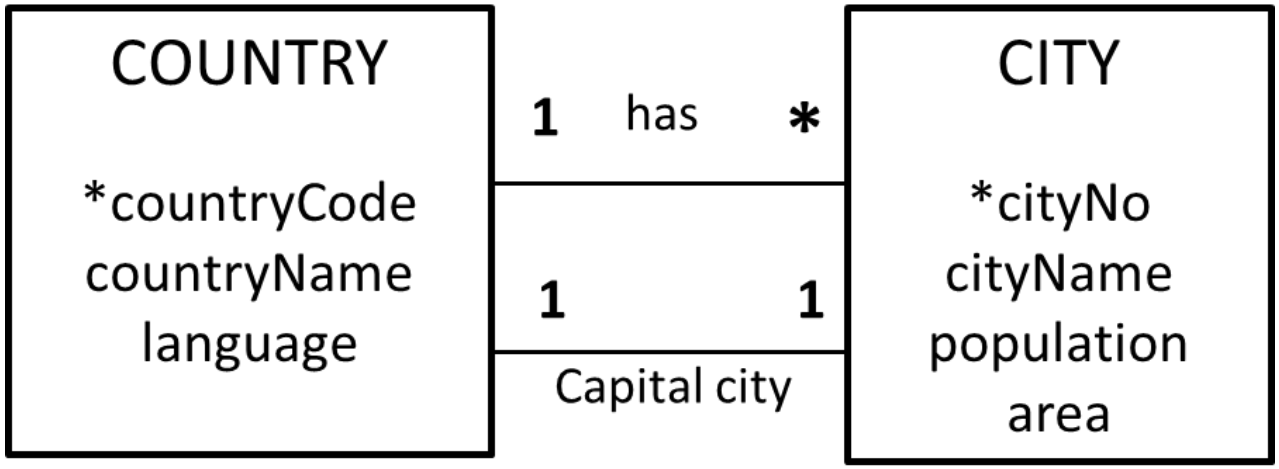
- B) Optional, Disjoint (or) 1 superclass and 6 subclasses

- C) Mandatory, Disjoint (or) 1 superclass and 4 subclasses

What have we learned?

- How to turn a conceptual ER model into a set of relations (relational model).
- How to place foreign keys (and new relations) to represent different multiplicities.
  - Different rules for mapping 1:1, 1:* *:* relationships
- How to represent subclass/superclass as a set of relations.
  - Different rules for optional/mandatory disjoint/nondisjoint

Exercise:
Derive relational schema from ER diagram



- This diagram shows the relationship between "City" and "Country".
- Derive a ***relational schema*** for the entities and relationships in the ER model below. Indicate primary key for each relation in the relational schema.

Exercise:
Derive relational schema from ER diagram