Database Systems

Instructor: Bilal Khalid Dar

Mapping from Data Model (ERD) to Relational Model

Transformation

- Transform the conceptual database design (ERD) into a logical database design that can be implemented on a chosen DBMS later (our choice: RDS)
  - Input: conceptual model (ERD)
  - Output: relational model (schema), normalized relations

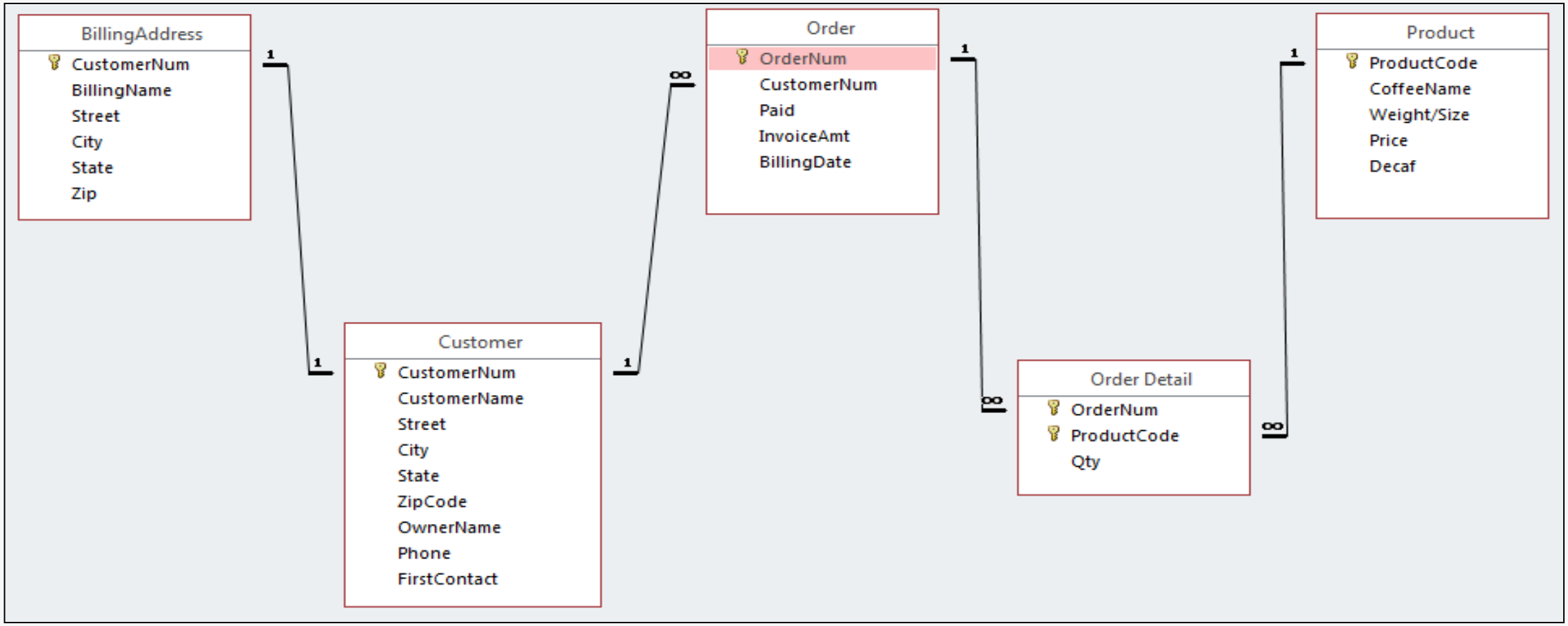
Why do I need to know this?

- CASE tools can perform many of the transformation steps automatically, but..
  - Often CASE tools cannot model complexity of data and relationship
    - Example: ternary relationship, supertype and subtype relationship
  - You must be able to perform a quality check on CASE tool results

Transformation Procedure

- An entity turns into a table.
- Each attribute turns into a column in the table.
- The (unique) identifier of the entity turns into a PK of the table.
- In general, the ERD (data model) does not include FK information.
  - Restaurant database: next slide

Access Data Model



Review of Relational Model (schema)

* The Relational model does not like any type of redundancies.

- Every table must have a unique name.
- Attributes in tables must have unique names.
- Every attribute value must be atomic.
- The order of the columns & rows are irrelevant.

Primary Key

- is a **key** in a relational database system that is unique for each record (or row) and a table as well.
  - Unique identifier of each record (and table).
- Example:
  - SS#, driver license number, vehicle identification number (VIN).

Rule for Primary Key

- No “null” value can be allowed.
  - Null value is not equal to zero.
  - OK to have “null” value for a non-PK.

	ID	LastName	FirstName	Number
1	1	Johnson	Joe	555-2323
2	2	Lewis	Larry	NULL
3	3	Thompson	Thomas	555-9876
4	4	Patterson	Patricia	NULL

Rule for Primary Key con't

- No duplication: No two CSUB students can have same ID number.
- PK can be "composite key"
  - More than one field (two, three, etc) can be defined as "Primary Key"
  - Example on the class website.
    - **"Composite PK Example"**
    - Employee ID + SS#, Student ID + SS#

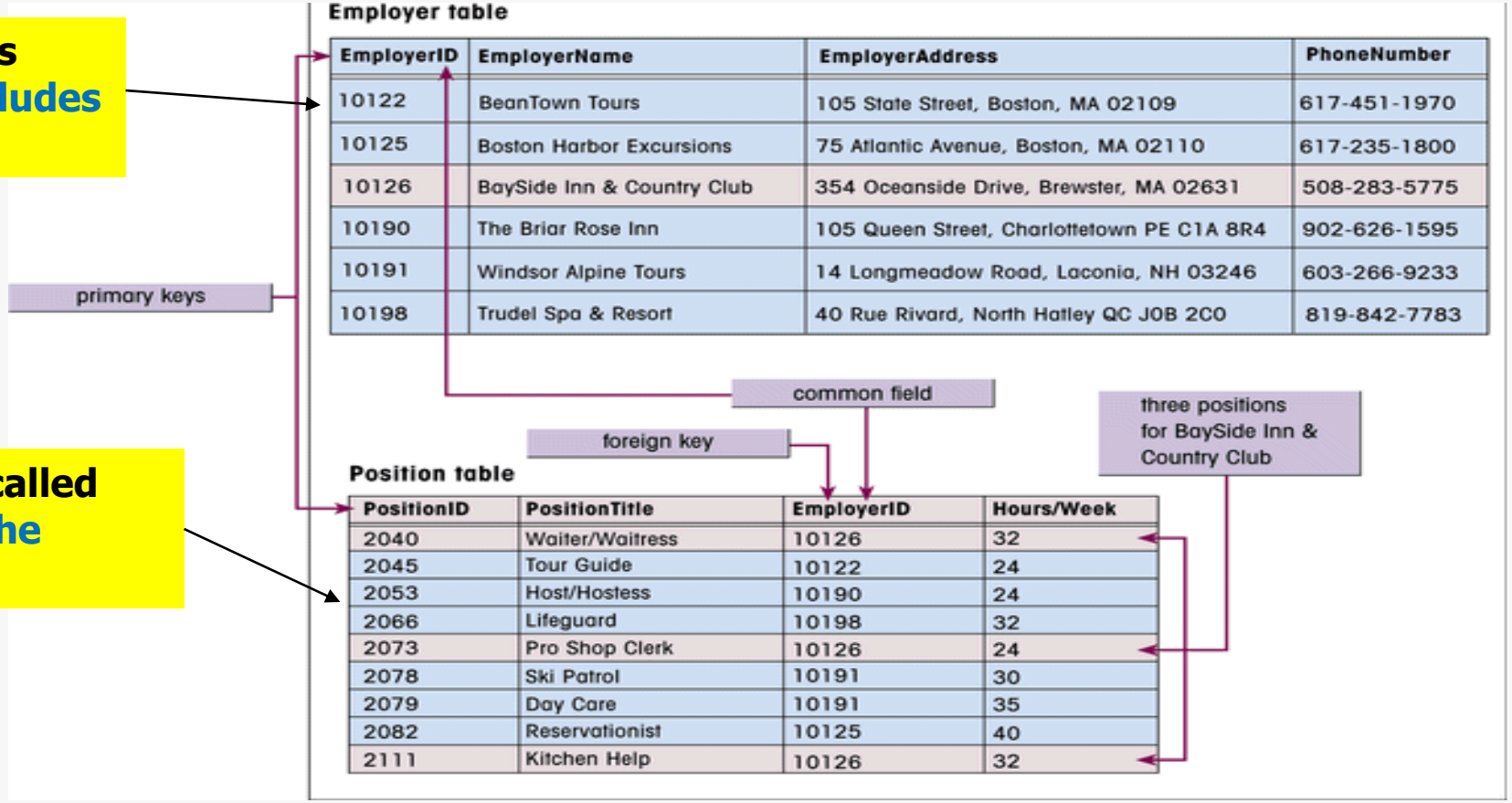
Foreign Key

- A foreign key is a field in one table that uniquely identifies a row of another table or the same table.
  - That is, the foreign key is defined in a second table, but it refers to the primary key in the first table.
  - In Access, the foreign key is defined in a second table (Related Table), but it refers to the Primary key in the first table (Primary Table).

Relating tables using PK and FK

Only on Access, the Employer table is called "Primary" table because it includes the primary key.

Only on Access, the Position table is called "Related" table. Because it includes the foreign key.



EmployerID	EmployerName	EmployerAddress	PhoneNumber
10122	BeanTown Tours	105 State Street, Boston, MA 02109	617-451-1970
10125	Boston Harbor Excursions	75 Atlantic Avenue, Boston, MA 02110	617-235-1800
10126	BaySide Inn & Country Club	354 Oceanside Drive, Brewster, MA 02631	508-283-5775
10190	The Briar Rose Inn	105 Queen Street, Charlottetown PE C1A 8R4	902-626-1595
10191	Windsor Alpine Tours	14 Longmeadow Road, Laconia, NH 03246	603-266-9233
10198	Trudel Spa & Resort	40 Rue Rivard, North Hatley QC J0B 2C0	819-842-7783

PositionID	PositionTitle	EmployerID	Hours/Week
2040	Waiter/Waitress	10126	32
2045	Tour Guide	10122	24
2053	Host/Hostess	10190	24
2066	Lifeguard	10198	32
2073	Pro Shop Clerk	10126	24
2078	Ski Patrol	10191	30
2079	Day Care	10191	35
2082	Reservationist	10125	40
2111	Kitchen Help	10126	32

Referential integrity

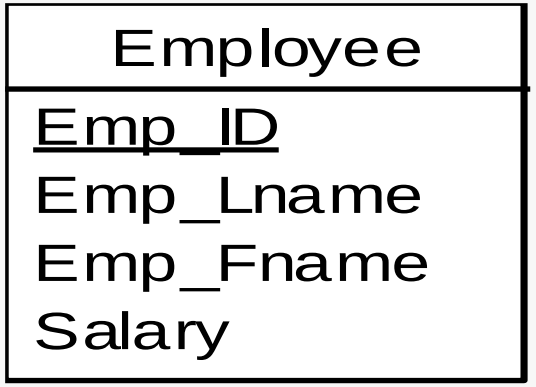
- Matching of primary and foreign keys
- Cascade delete
- Cascade Update
- Assign default value (e.g., 999)
- Set to null

Composite and Multi-valued Attributes

- An attribute is considered composite if it comprises two or more other attributes.
  - Divided into atomic and separate attribute
- There should no such thing as a multi-valued attribute (phone #) in a relational database.
  - Against “Set Theory”
  - “Multi-valued Attribute” must be turned into a new entity of its own…..

Mapping an entity into a relation

- An Entity name: Employee
- Attributes:
  - Emp_ID, Emp_Lname, Emp_Fname, Salary
- Identifier: Emp_ID



Employee

Emp_Id PK	Emp_Lname	Emp_Fname	Salary

Mapping an entity into a relation

Movies
Title Year Length Film Type

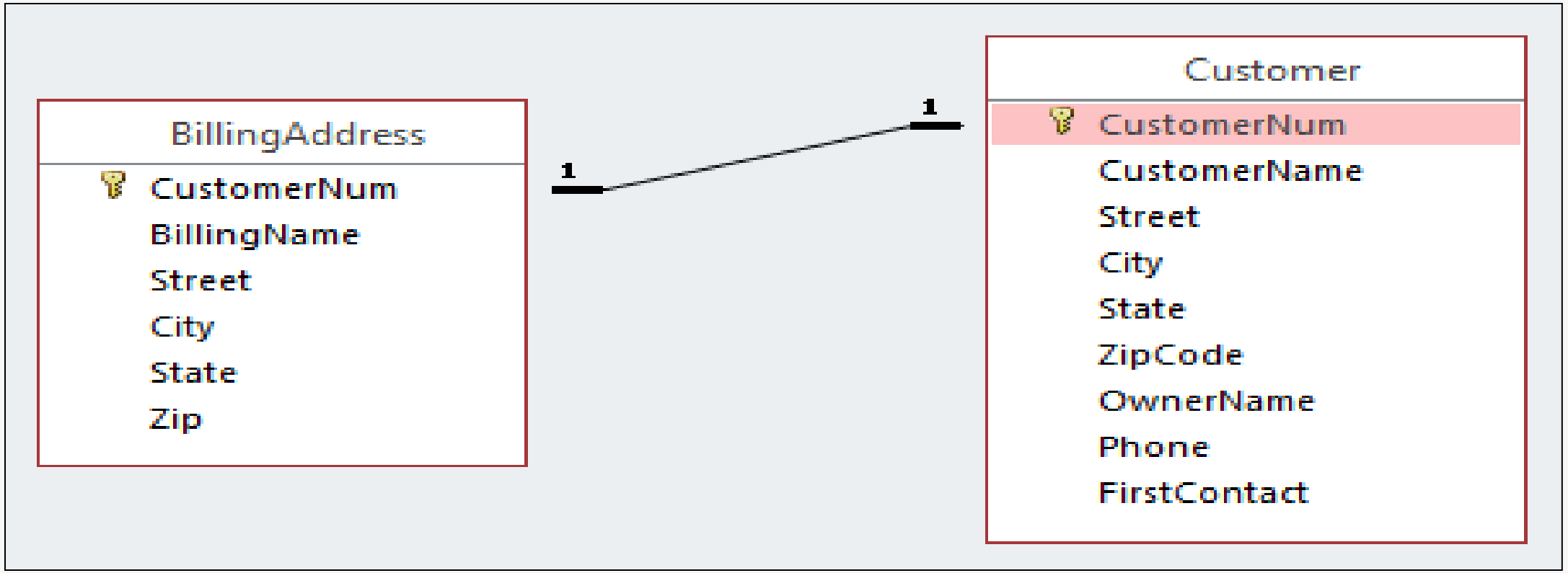
Movies

title	***year***	*length*	*filmType*
Star Wars	1977	124	color
Mighty Ducks	1991	104	color
Wayne's World	1992	95	color

Mapping binary relationships

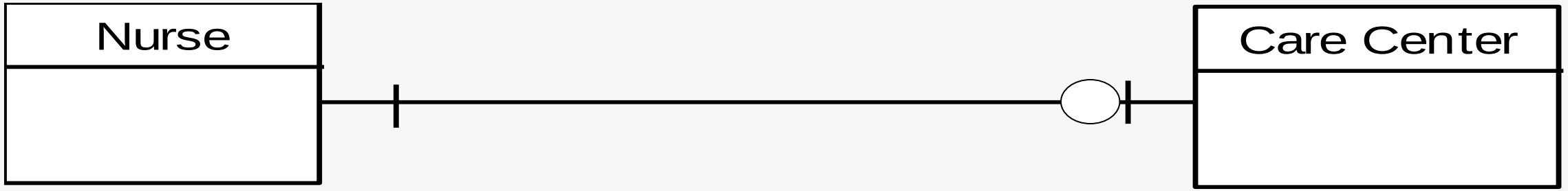
- One-to-one: if there is no indication of optionality, then it needs to be decided by developers.
  - one-to-one mandatory relationship
  - Restaurant DB: BillingAddress and Customer
- One-to-many: PK on the one side becomes FK on the many side
- Many-to-many - create a new relation (bridge entity) with the PKs of the two entities as its composite PK

Mapping a 1:1 relationship

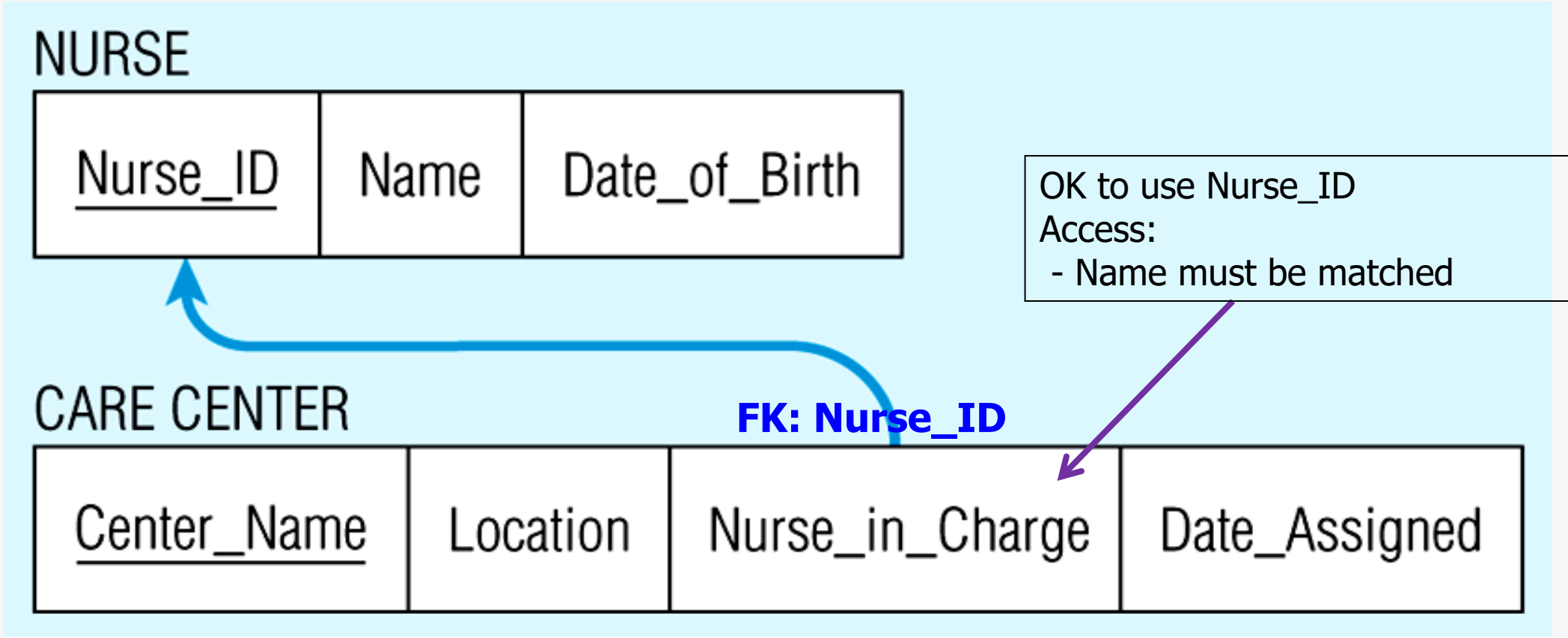


Mapping a 1:1 relationship with optionality on the one side

- Nurse:
  - **Nurse_ID**, Name, Date_of_Birth
- Care Center
  - **Center_Name**, Location, Date_Assigned



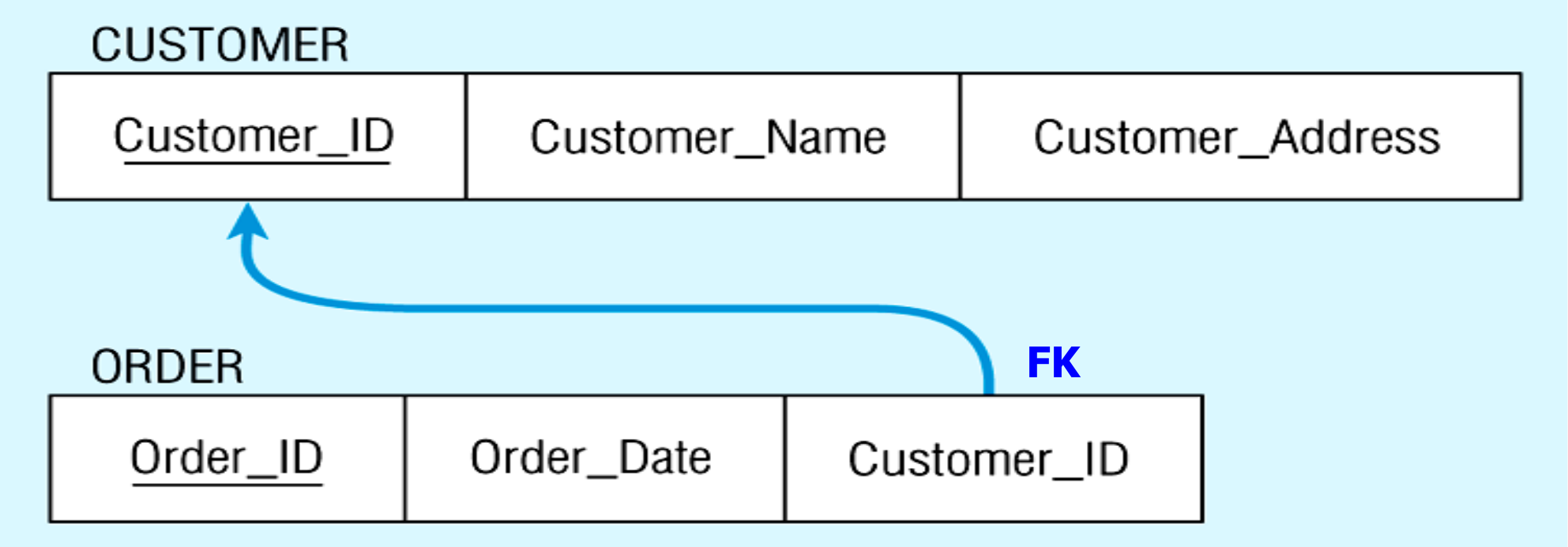
Mapping a 1:1 relationship



Mapping a 1:M relationship

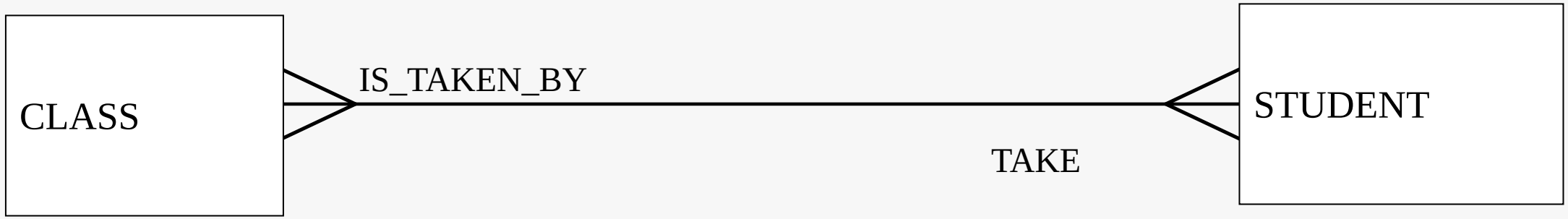
- Customer:
  - **Customer_ID**, Customer_Name, Customer_Address
- Order:
  - **Order_ID**, Order_Date

Mapping a 1:M relationship



Mapping M:N relationship

Each student takes many classes, and a class must be taken by many students.



Transformation of M:N

- The relational operations become very complex and are likely to cause system efficiency errors and output errors.
- Break the M:N down into 1:N and N:1 relationships using bridge entity (weak entity).

Example M:N Relationship

Database name: CH2_TEXT

Table name: STUDENT_FIG2_24

STU_NUM	STU_LNAME	CLASS_CODE
321452	Bowser	10014
321452	Bowser	10018
321452	Bowser	10021
324257	Smithson	10014
324257	Smithson	10018
324257	Smithson	10021

Table to represent Entity

3 to 3
30 to 30
300 to 300
3000 to 3000
30,000 to 30,000
300, 000 to 300, 000

Table name: CLASS_FIG2_24

CLASS_CODE	STU_NUM	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	321452	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10014	324257	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	321452	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10018	324257	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	321452	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10021	324257	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

Converting M:N Relationship to Two 1:M Relationships

Table name: STUDENT_FIG2_25
Primary key: STU_NUM
Foreign key: none

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

Table name: ENROLL_FIG2_25
Primary key: CLASS_CODE+STU_NUM
Foreign key: CLASS_CODE, STU_NUM

CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Bridge Entity

Table name: CLASS_FIG2_25
Primary key: CLASS_CODE
Foreign key: CRS_CODE

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

Mapping an M:N relationship

Student

STU_NUM	STU_LNAME

Enroll (added later!)

CLASS_CODE	STU_NUM	ENROLL_GRADE

Class

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME

CLASS_ROOM	PROF_NUM

Mapping a bridge entity

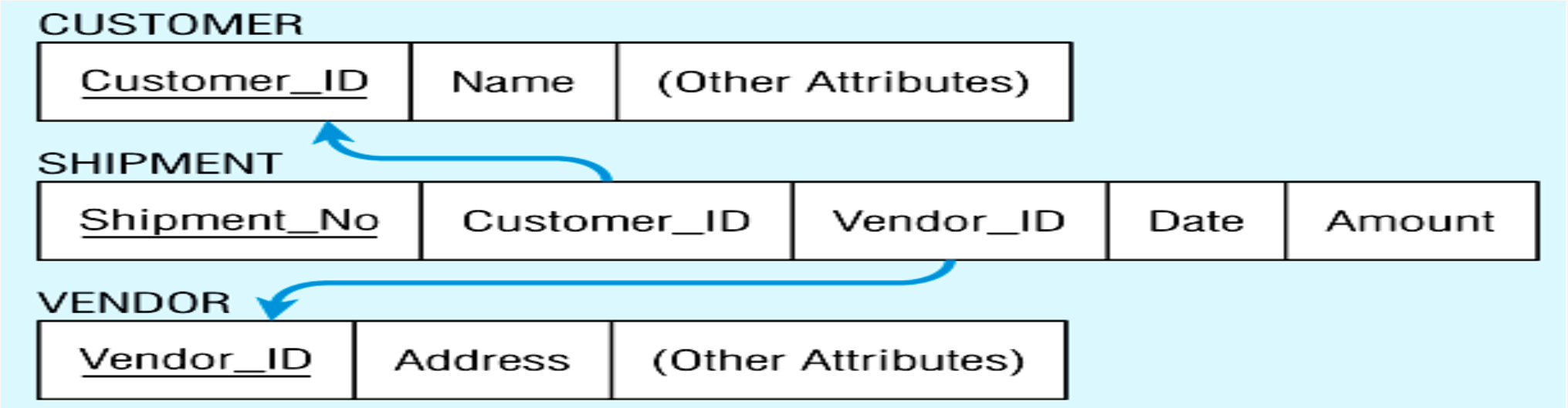
with its own name and identifier

Customer_ID
Name
Other Attributes

Shipment_NO
Customer_ID
Vendor_ID
Date
Amount

Vendor_ID
Address
Other Attributes

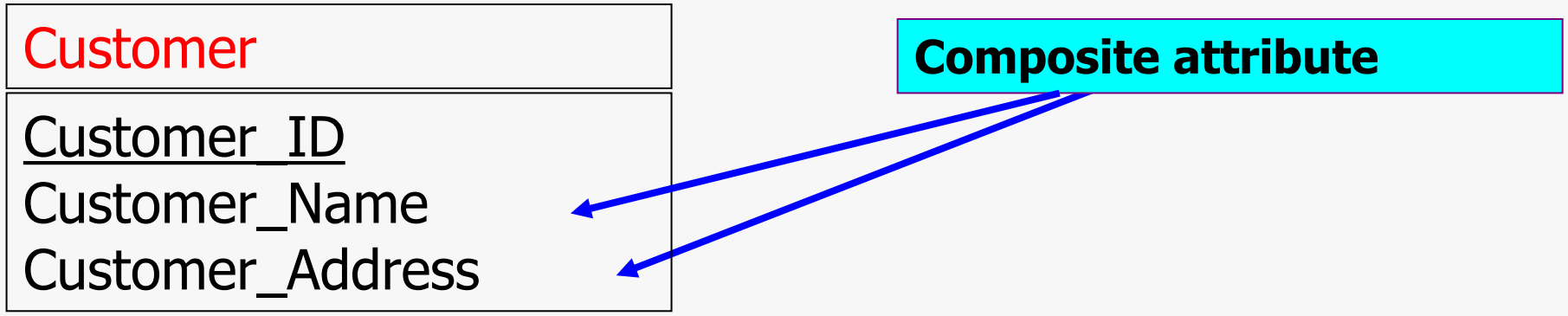
Mapping a bridge entity
with its own identifier (con't)



Mapping composite and Multi-valued attributes to relation

- Composite attribute: use only their simple, component attributes – divide into atomic and separate attribute.
- Multi-valued attribute: turned it into a new entity of its own…..

Mapping composite attributes to relation



CUSTOMER

Customer_ID	Customer_Name	Street	City	State	Zip

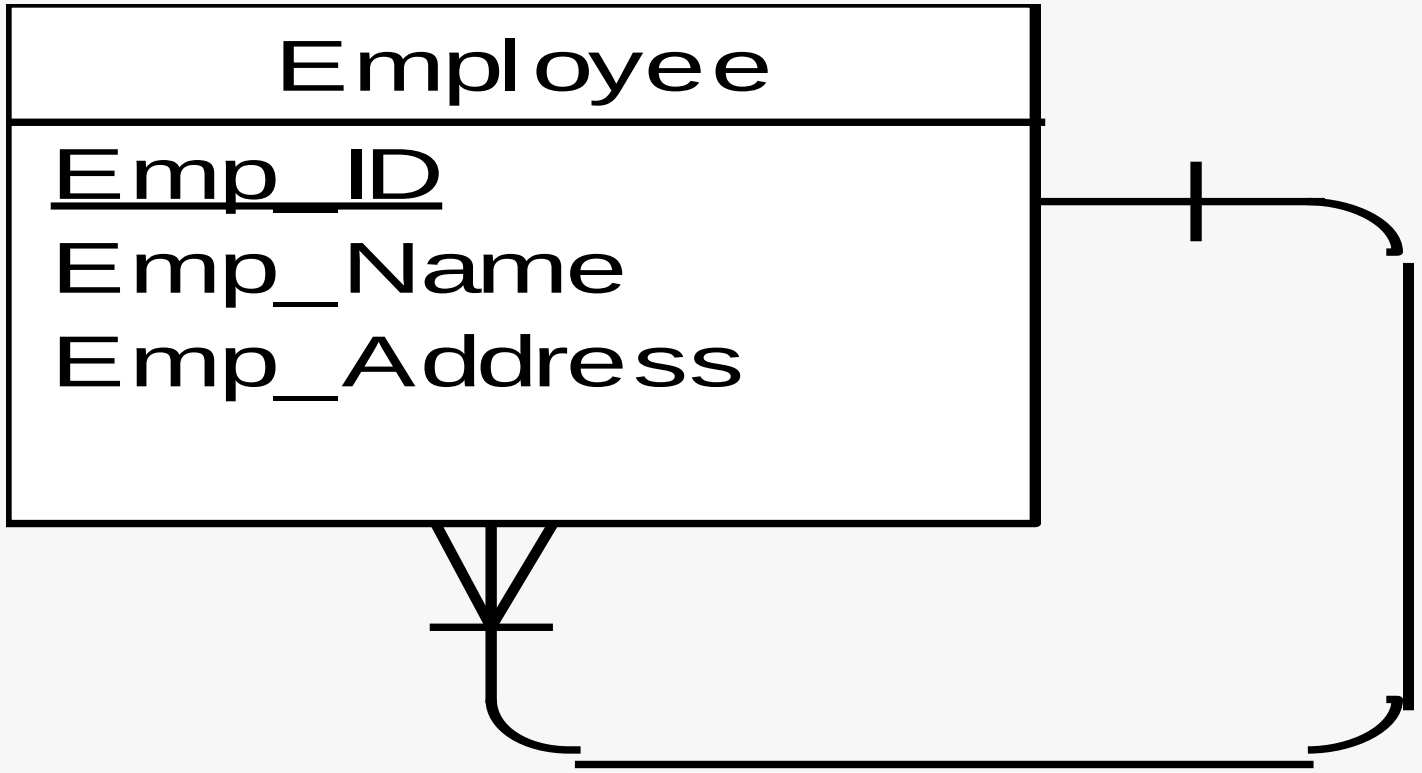
Mapping a multi-valued attribute

Employee
SSN Name **Phone #**

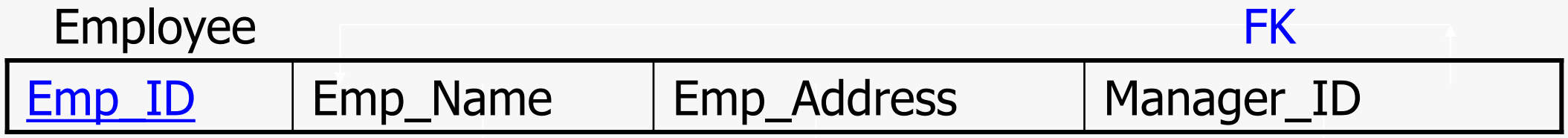
Employee	
SSN	Name
E101	Johnson
E102	Smith
E103	Conley
E104	Roberts

Phone	
SSN	Phone#
E101	312 …
E102	708 …
E102	312 …
E104	603 …

Mapping 1:M recursive (or unary) relationship



Mapping 1:M recursive (or unary) relationship

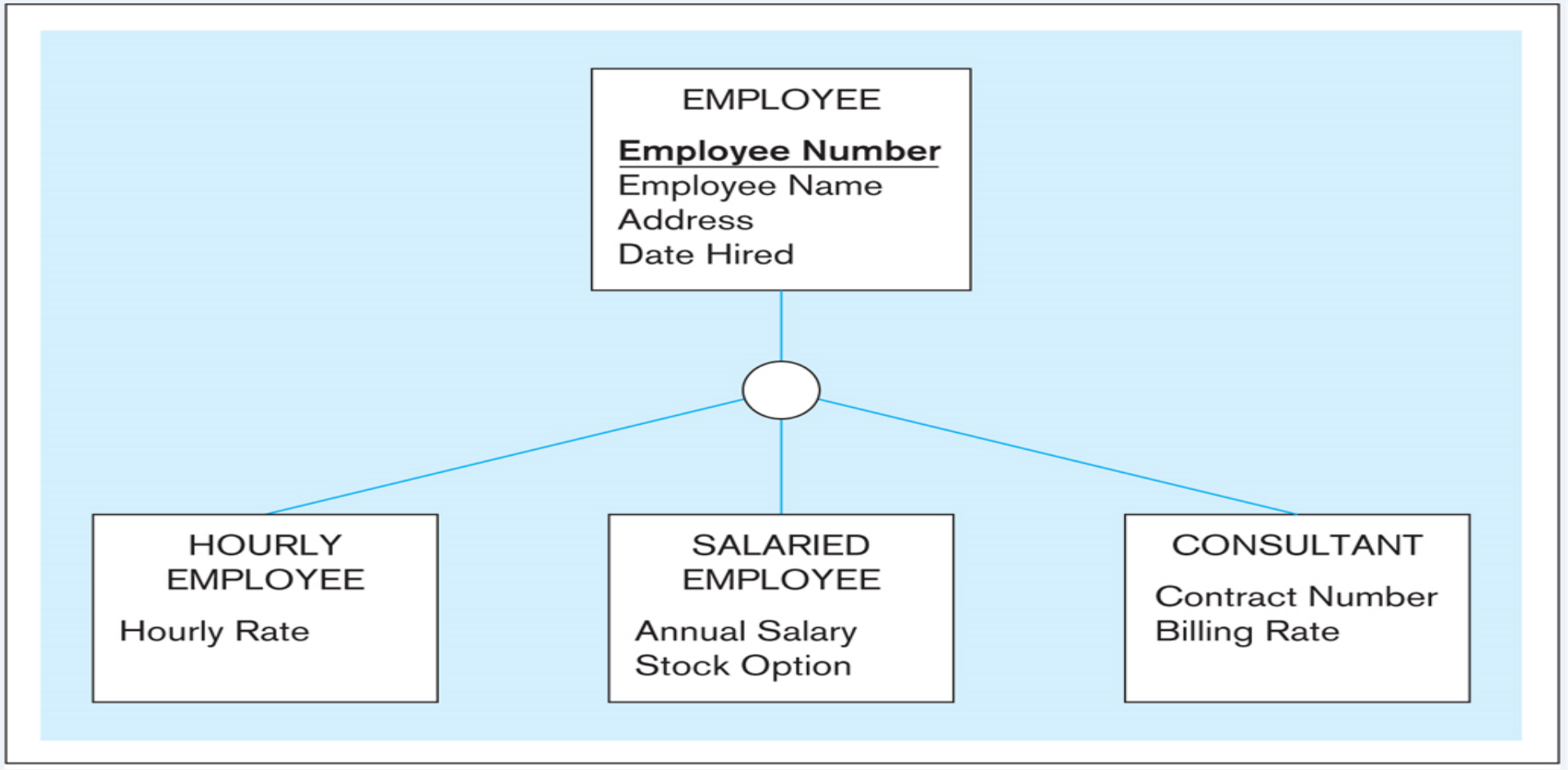


- Manager_ID references Emp_ID

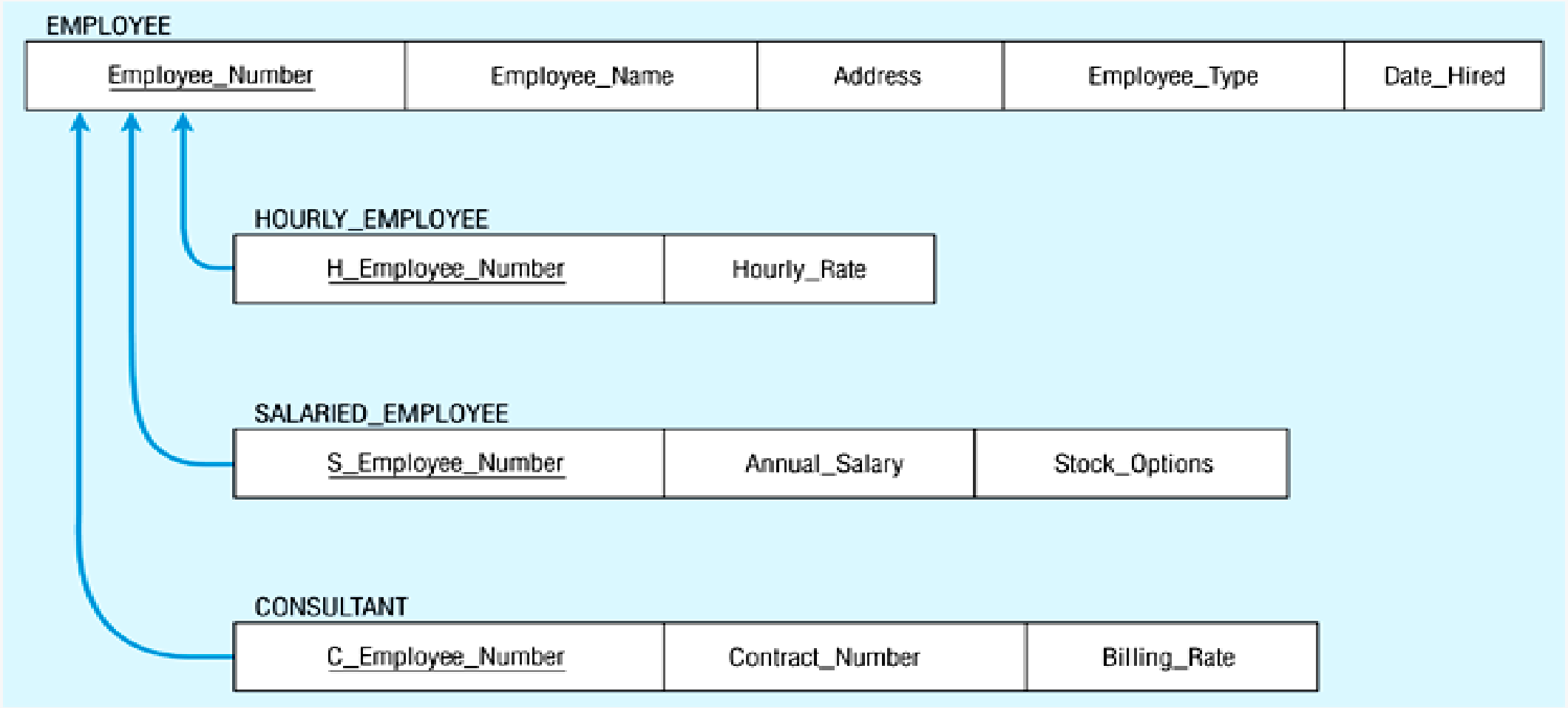
Mapping Supertype/subtype relationship

- Create a separate relation for the supertype and each of the subtypes
- Assign common attributes to supertype
- Assign PK and unique attributes to each subtype
- Assign an attribute of the supertype to act as subtype discriminator

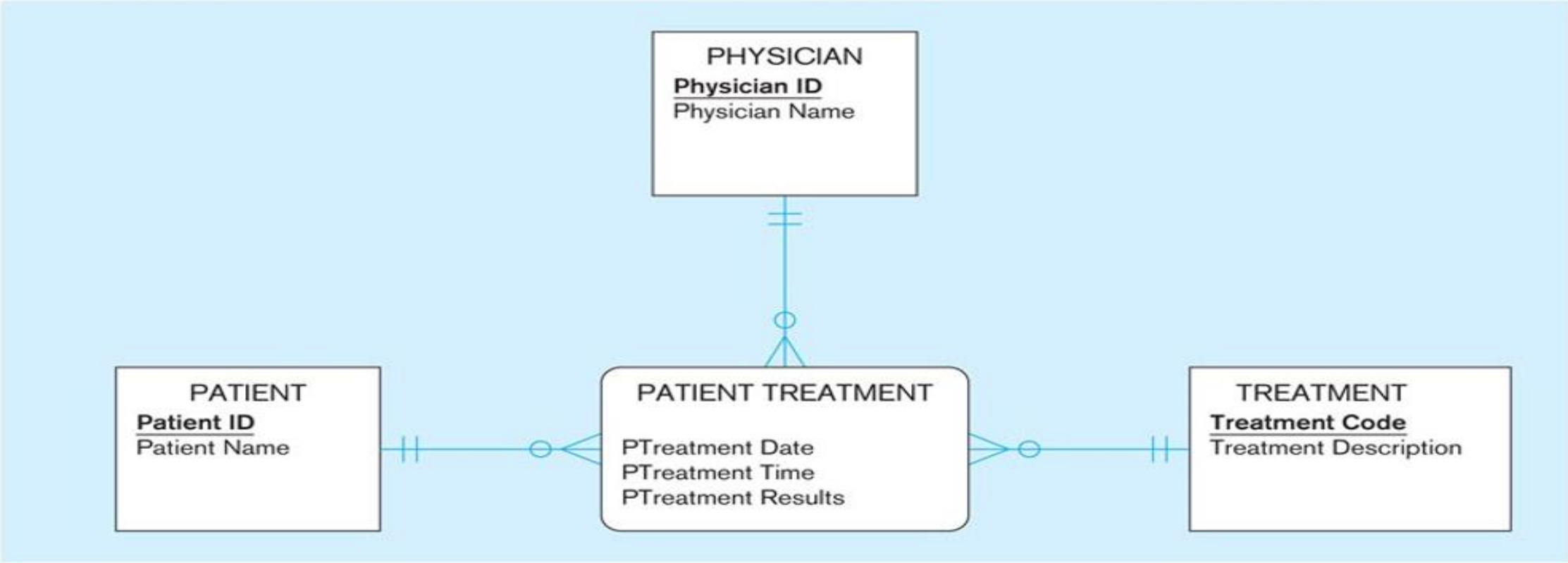
Mapping Supertype/subtype relationship



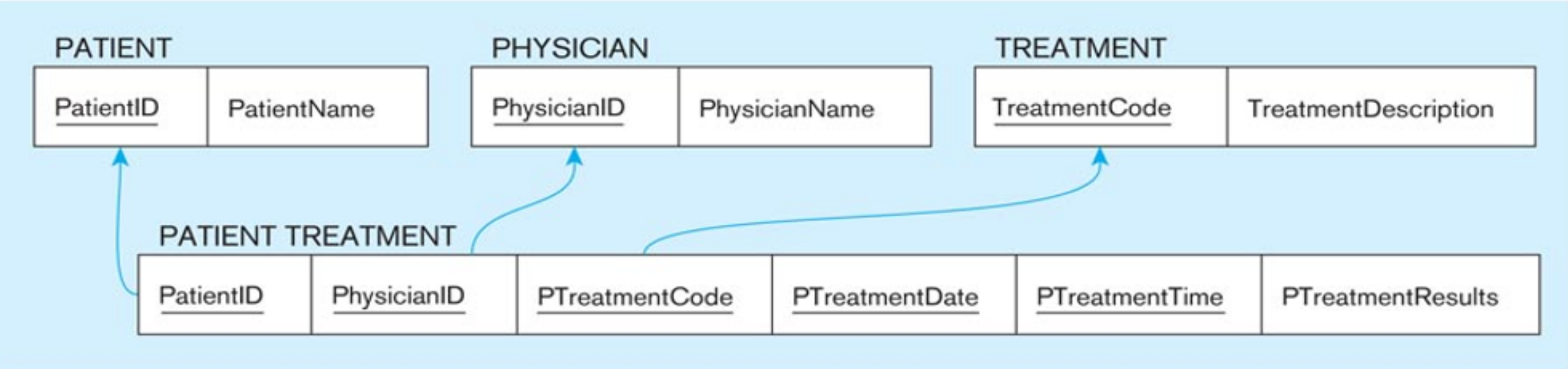
Mapping Supertype/subtype relationship



Mapping Ternary Relationship



Mapping Ternary Relationship




Thank
You