ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

Database Systems

Chapter 2: Database Design

session 2:
Relational Data Model

http://vku.udn.vn/

Outline

1 Relational Data model

2 ER to Relational Mapping

Relational Database

❑Basic structure:

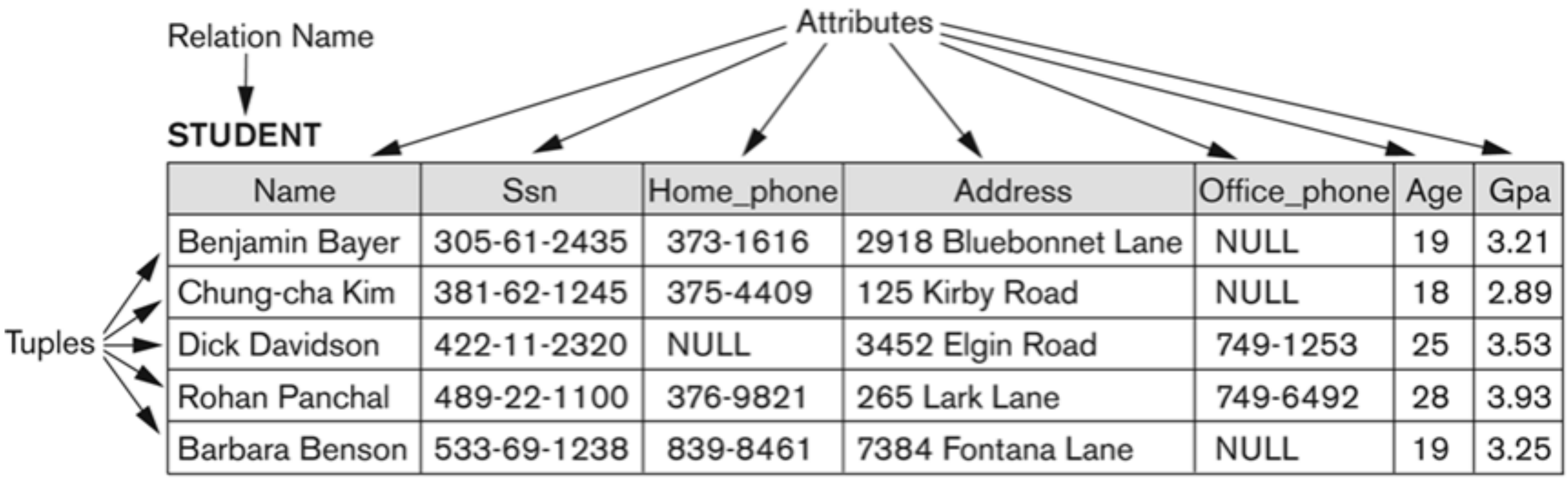
- A relational database consists of a collection of **tables**, each of which is assigned a unique name.
- Tables are also known as *Relation*.
- The columns in a relation are known as **attributes**. The order of attributes is insignificant
- A **tuple** is a row of a relation. They are also called the records.
  - Each tuple is unique.
  - The order of tuples is insignificant
- Tables are related to each other through some shared attributes.

Relational Database

- The set of allowed values for each attribute is called the **domain** of the attribute
- Attribute values are (normally) required to be **atomic**; that is, indivisible
- A **null** value is a special value that signifies that the value is *unknown* or *does not exist*.
- The null value causes complications in the definition of many operations

Relational Database

- ❑Example: **Student** Relation

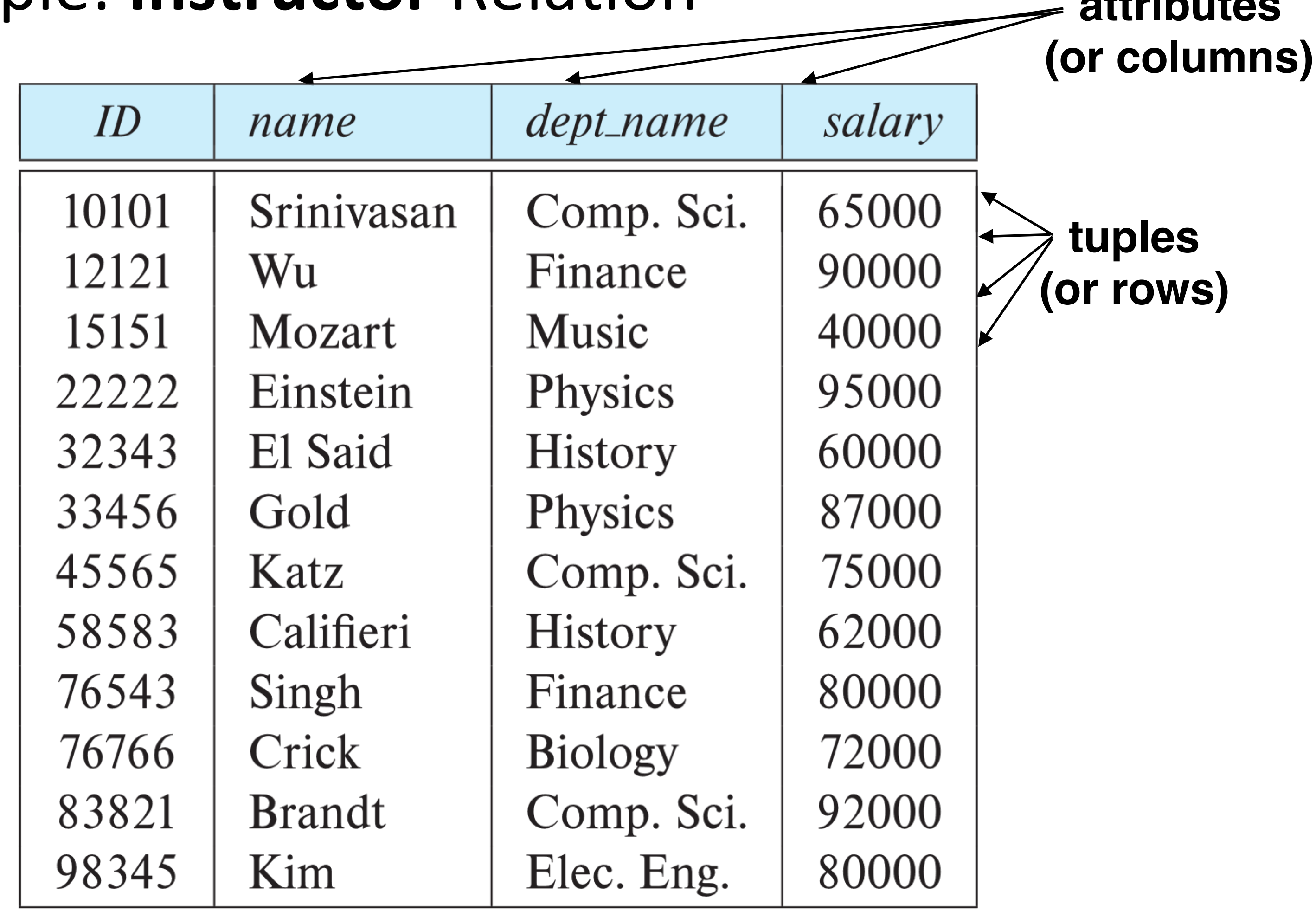

Relation Name
STUDENT
Attributes
Tuples

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

The attributes and tuples of a relation STUDENT.

Relational Database

- Example: **Instructor** Relation



ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

attributes (or columns)

tuples (or rows)

Relation Scheme

❑It is a named of a relation defined by a set of attributes and their corresponding domains.

❑Common convention:

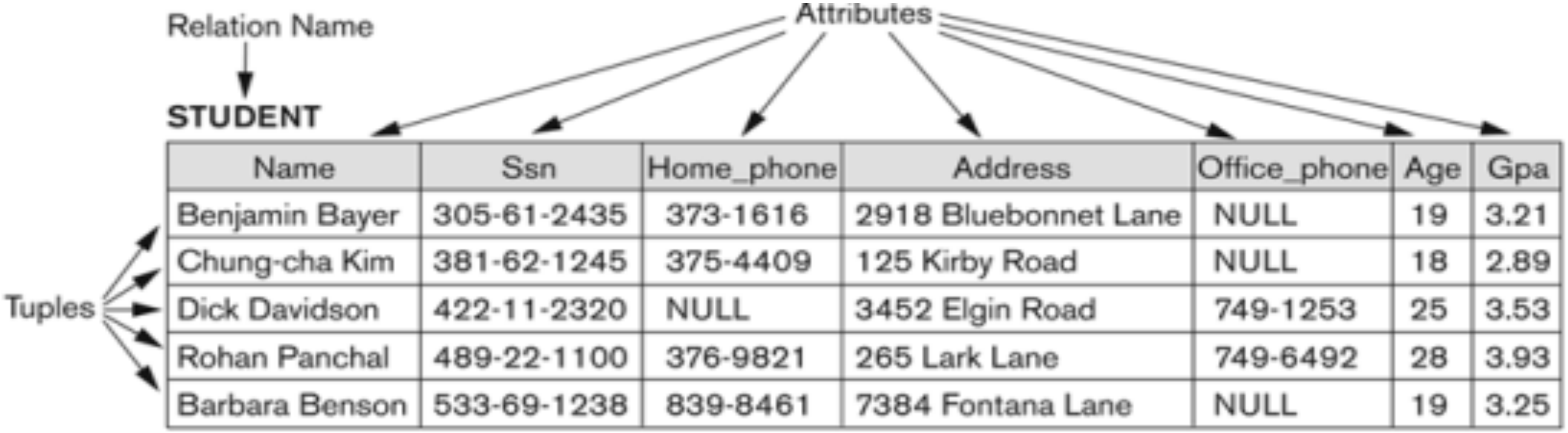
- RelationName (attribute1, attribute2,….,attribute_n)
- The primary key is underlined.

❑Example:

- Instructor (ID, name, dept_name, salary)
- Branch(branchNo, street, city, postcode)

Relation Schema

❑ Characteristics of relation scheme

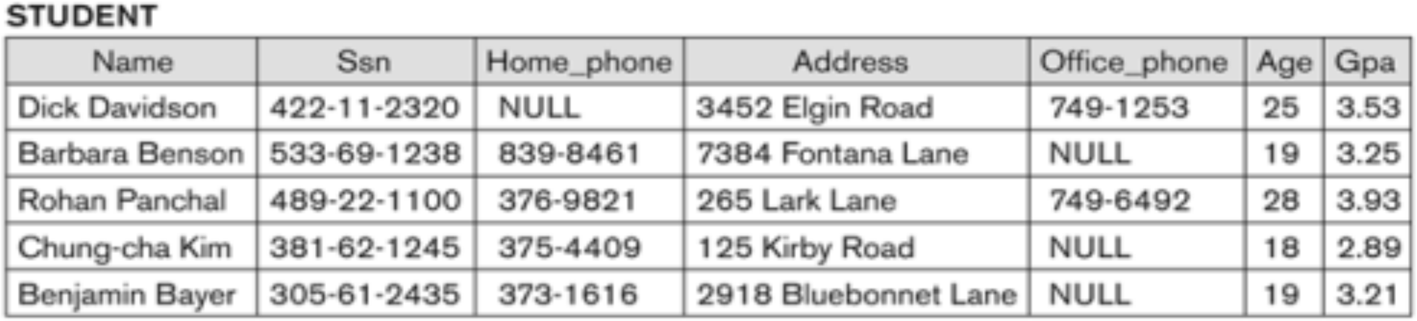


STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

The attributes and tuples of a relation STUDENT.

The relation STUDENT with a different order of tuples.



STUDENT

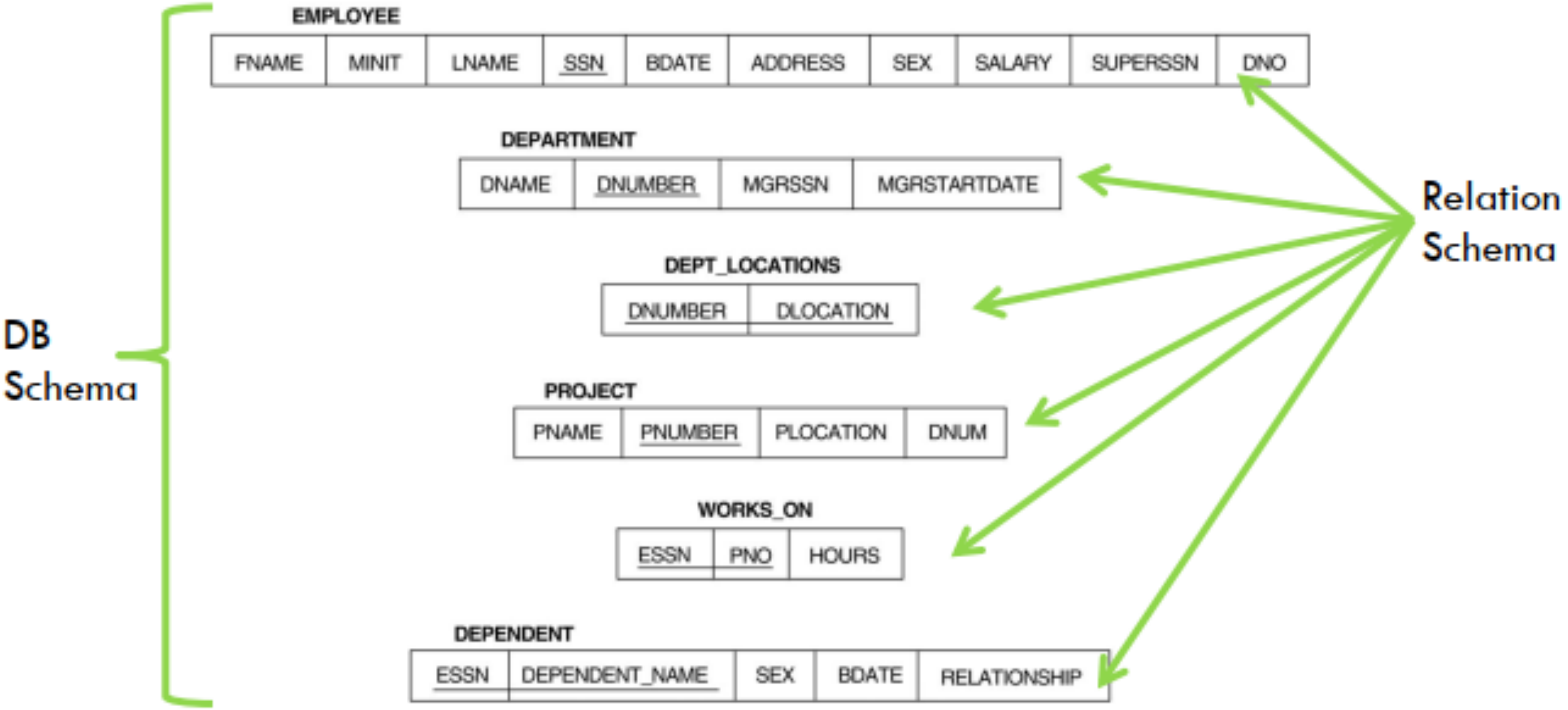
Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21

Different order of tuple don't have any significance

Relational Database Schema

- ❑ It is a sets of relation schema.
- ❑Example: the relational database schema for

COMPANY = {EMPLOYEE, DEPARTMENT, DEPENDENT, PROJECT}



EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

Relational Database Schema

❑ The relation schema normally represented as follows:

EMPLOYEE (SSN, FNAME, LNAME, MINIT, BDATE, ADDRESS, SEX, SALARY, SUPERSSN,DNO)
DEPARTMENT (DNUM, DNAME, MGRSSN, MGRSTTDATE)

❑ **Relation instance**

- Is a *tuple* at a specific moment of time
- Eg: Branch (BranchNo, Street, City, PostalCode)
  The relation instance for branch is:
  - ✓(B005, 55 Jln Dobi, Johor Bahru, 80100)
  - ✓(B006, 55 Jalan Perai, Johor Bahru, 80000)
- The relation instance change when tuple is updated, deleted or inserted.

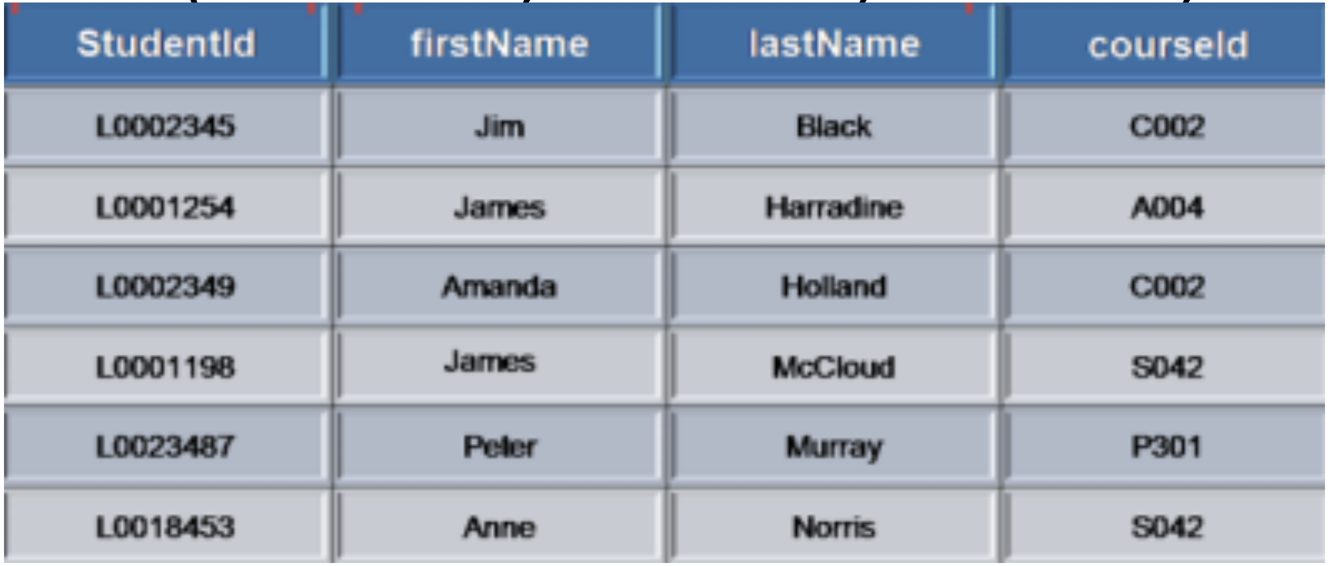
Relation Keys

- ❑Refers to the important attribute in an entity.
- ❑Determine the uniqueness of an row in given table.
- ❑Identifiers for each rows.
- ❑An attribute or more than one attributes can be declared as keys depending on situations.
- ❑Types of keys:
  - ▪Superkey
  - ▪Candidate Key
  - ▪Alternative key
  - ▪Primary Key
  - ▪Foreign key

Superkey

❑**Superkey** is a an attribute, or set of attributes which can uniquely identify a tuple in a relation.

Example: Student(StudentId, firstName, lastName, courseId)



StudentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	James	McCloud	S042
L0023487	Peler	Murray	P301
L0018453	Anne	Norris	S042

The super key can be any of the following:

- ✓StudentId
- ✓StudentId, lastName
- ✓StudentId, firstName, lastName

Candidate Key

- ❑ A candidate key is a **minimal** superkey. A superkey that does not contain a subset of attributes that is itself a superkey. This means you cannot remove any fields from candidate key, else it will not be able to uniquely identify the rows.
- ❑There can be more than one candidate key in a table
- ❑When a key consists of more then one attribute it is known as the **composite key**
- ❑Example: The candidate key for Student relation can be any of the following:
  - ✓StudentId
  - ✓StudentId, lastName

Alternative Key

❑An alternate key is any candidate key that is not primary key.

❑Alternate keys are sometimes referred as secondary keys

❑The secondary key is defined as a key that is used strictly for data retrieval purposes

Primary key

- ❑ A primary key is one of the candidate keys selected to uniquely identify a tuple in a relation.
- ❑ Which one?
  - ▪ Primary keys must be chosen with care. Ex: name, SSN, ID, …
  - ▪ The primary key should be chosen such that its attribute values are never, or are very rarely, changed. Ex: address, mobile,…
- ❑ Each table must have primary key.
- ❑ Cannot be NULL value to maintain Entity Integrity.
- ❑ Example: for Student relation
  - ▪ StudentId can be chosen to be the primary key

Foreign Key

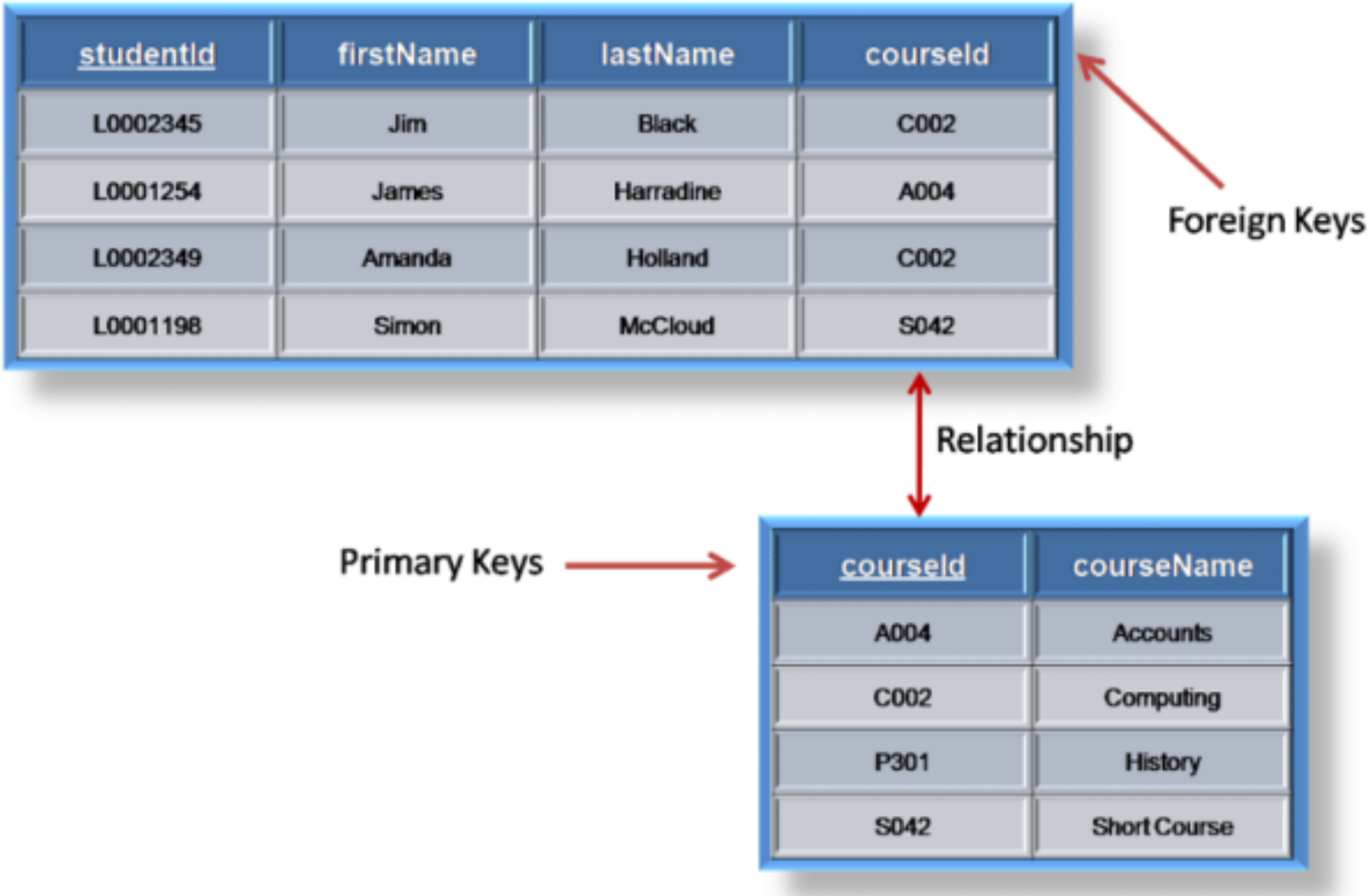
- ❑A foreign key is an attribute (or a set of attributes) whose values match the primary key values in related relation.
  - ▪Referencing relation
  - ▪Referenced relation
- ❑Example

Course(***courseId***, *courseName*)
Student(**Student*Id***, first*Name*, *lastName*, *courseId*)

- The attribute *courseId* is the primary key of *Course*
- The attribute *courseId* in *Student* relation is a foreign key
- Student relation is called referencing relation of the foreign key constraint.
- Course relation is called the referenced relation

Foreign Key

- Example



studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042

Foreign Keys

Relationship

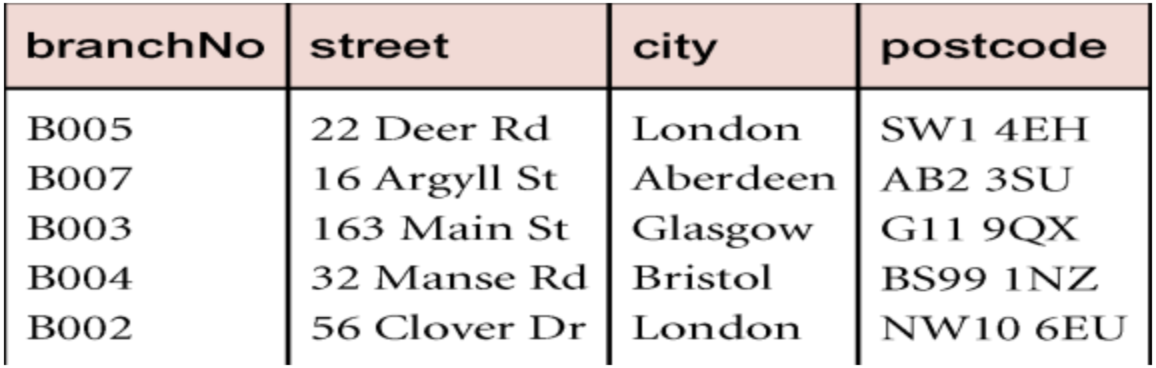
Primary Keys

courseId	courseName
A004	Accounts
C002	Computing
P301	History
S042	Short Course

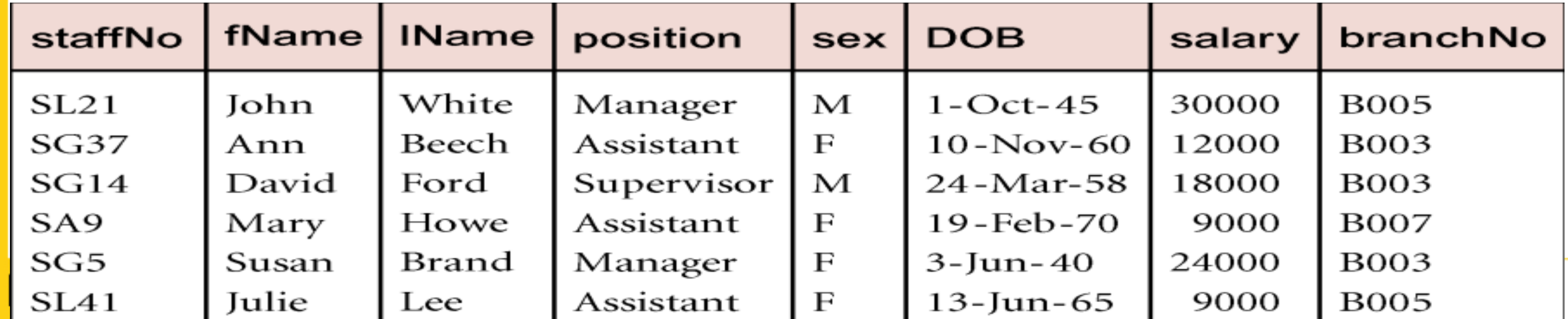
Foreign Key

❑Another Example

- Branch (branchNo, street, city, postCode)
- Staff (staffNo, fName, lName, position, sex, DOB, salary, branchNo)
- Staff has a Foreign Key is branchNo references Branch (branchNo)



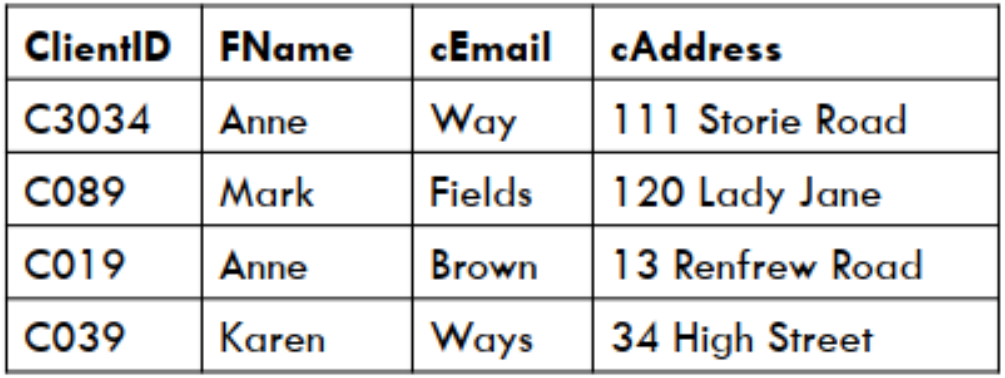
branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU



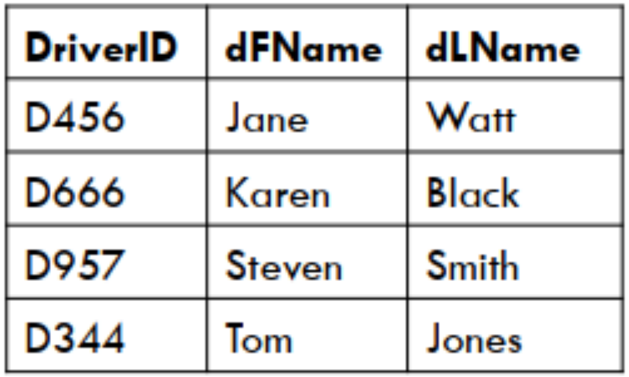
staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Relation Keys

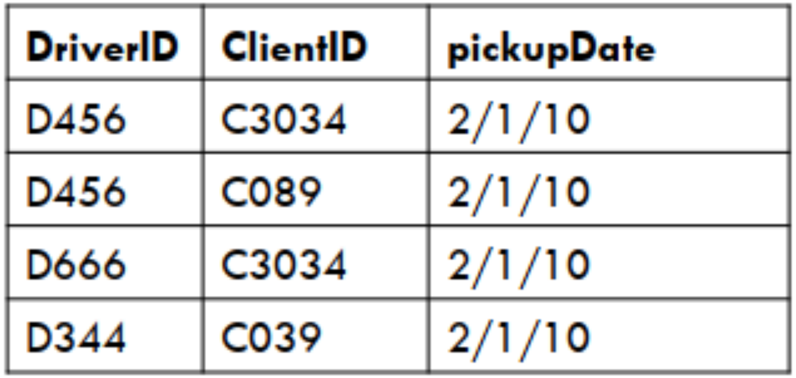
Group discussion



ClientID	FName	cEmail	cAddress
C3034	Anne	Way	111 Storie Road
C089	Mark	Fields	120 Lady Jane
C019	Anne	Brown	13 Renfrew Road
C039	Karen	Ways	34 High Street



DriverID	dFName	dLName
D456	Jane	Watt
D666	Karen	Black
D957	Steven	Smith
D344	Tom	Jones



DriverID	ClientID	pickupDate
D456	C3034	2/1/10
D456	C089	2/1/10
D666	C3034	2/1/10
D344	C039	2/1/10

For each table, find:

1. Two candidate keys
2. Primary Key
3. Foreign Key

Integrity rules

- ❑To have a good design, a database must have integrity rules.
- ❑Constraint or restriction that apply to all instances of the database.
- ❑Integrity rules consists of
  - ▪ Entity Integrity
  - ▪Referential Integrity

Entity Integrity

❑ Requirement

- All *Primary Key* entries are unique, and no part of a primary key may be ***NULL***.

❑ Purpose

- Each row will have a unique identity, and foreign key values can properly reference primary key values

❑ Example

- In the **Employee** table, EmpNo is the primary key, it can not have a duplicate number (All employees are UNIQUELY identified by their EmpNo number). And it can not be NULL.
- The **OrderDetail** has a composite primary key OrderNo and ProductNo so to insert a new row both values must be known.

Entity Integrity

❑ Other integrity rules that can be enforced in the relational model are the *NOT NULL* and *UNIQUE* constraints.

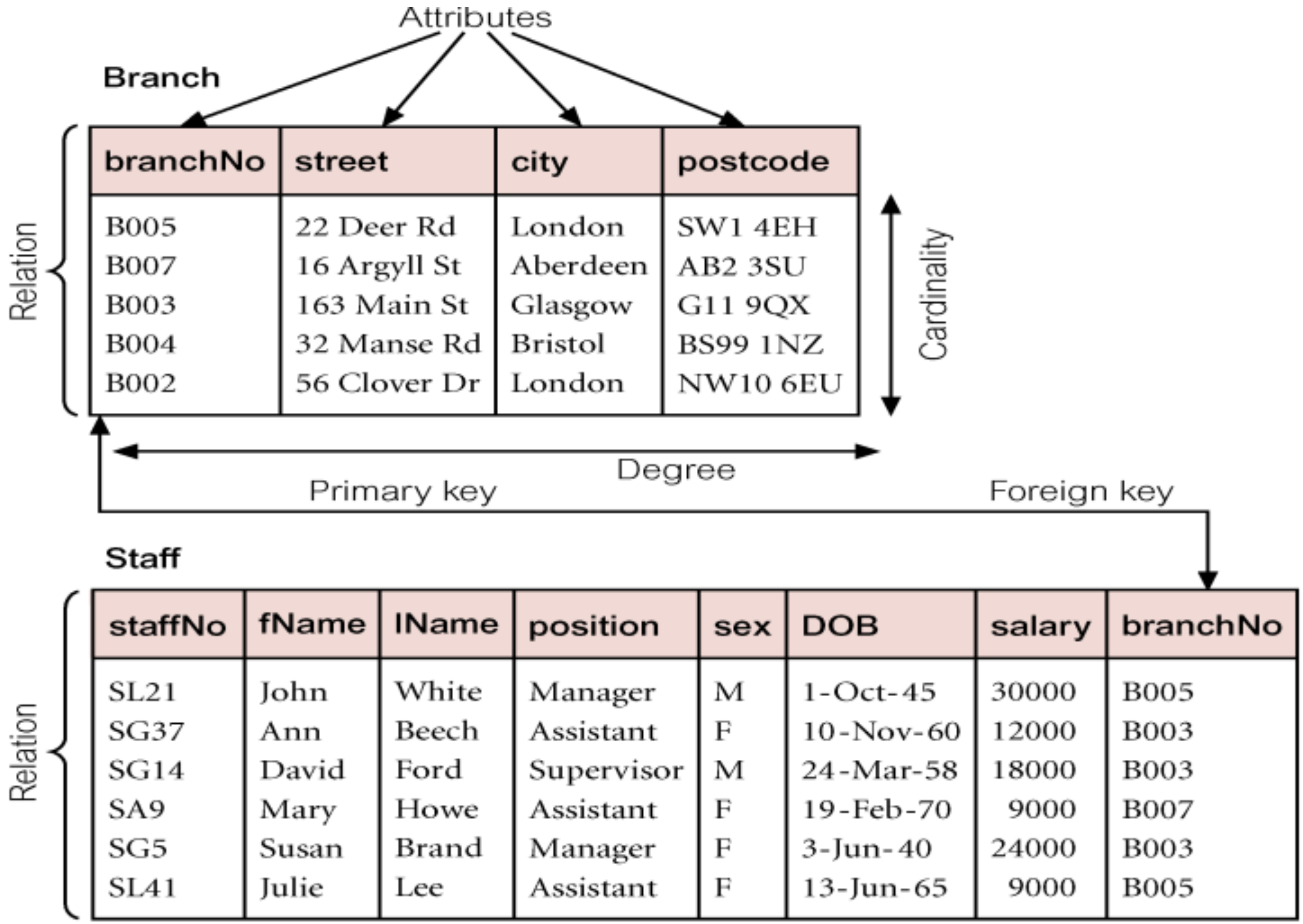
- The NOT NULL constraint can be placed on a column to ensure that every row in the table has a value for that column.
- The UNIQUE constraint is a restriction placed on a column to ensure that no duplicate values exist for that column

Referential Integrity

- ❑ Requirement
  - ▪ Every non-null foreign key value must reference an existing primary key value in the referenced relation. Or
  - ▪ The foreign key value can be null.
- ❑ Purpose
  - ▪ Makes it possible for an attribute NOT to have a corresponding value, but will be impossible to have an invalid entry.
  - ▪ The enforcement of the referential integrity rules makes it impossible to delete a row in one table whose primary keys has mandatory matching foreign key values on another table.

Referential Integrity

- ❑ Example: Branch and Staff Relation.



Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Referential Integrity

❑ Example: Branch and Staff Relation.

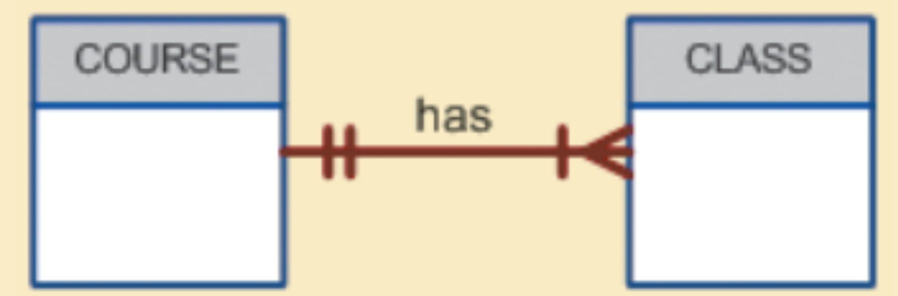
- It is not possible to create a staff record in Staff Relation with branchNo B025, unless there is already record for branch B025 in Branch relation.
- However, we should be able to create new staff record with NULL branch number to allow the situation where a new member staff has joined the company but has not yet assigned to a particular Branch.

Relationships within the Relational Database

- ❑**The 1:M relationship** is the relational modeling ideal. Therefore, this relationship type should be the norm in any relational database design.
- ❑**The 1:1 relationship** should be rare in any relational database design.
- ❑**M:N relationships** cannot be implemented as such in the relational model. Later in this section, you will see how any M:N relationship can be changed into two 1:M relationships.

1-M Relationship

- ❑Example: The ERM’s 1:M relationship between COURSE and CLASS
  - Each COURSE can have many CLASSes, but each CLASS references only one COURSE



1-M Relationship

- The implemented 1:M relationship between PAINTER and PAINTING in relational database

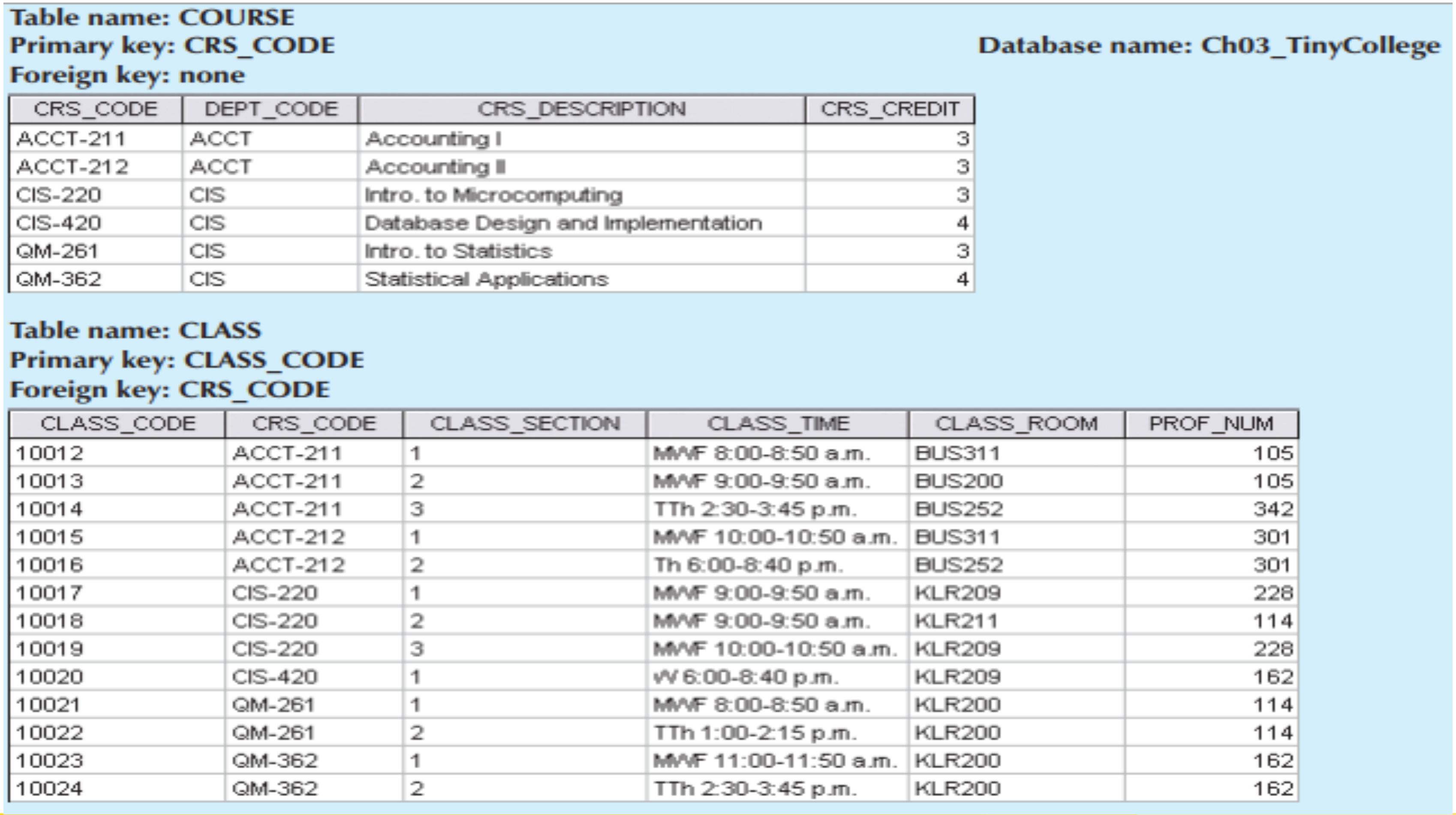


Table name: COURSE
Primary key: CRS_CODE
Foreign key: none

Database name: Ch03_TinyCollege

CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

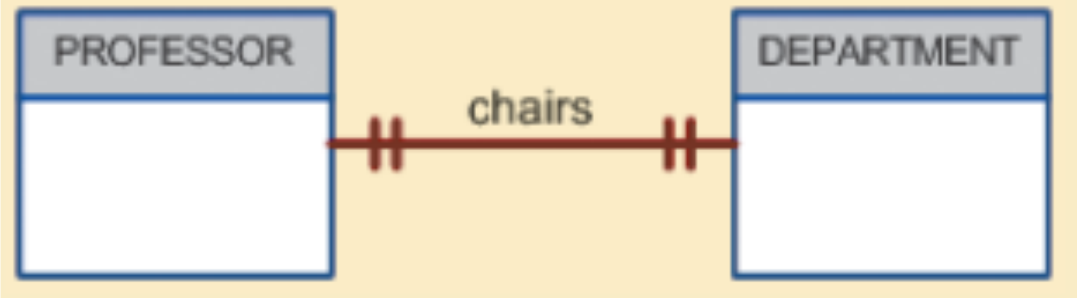
Table name: CLASS
Primary key: CLASS_CODE
Foreign key: CRS_CODE

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10012	ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
10013	ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10015	ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
10016	ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
10017	CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10019	CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
10020	CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10022	QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
10023	QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
10024	QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162

1-1 Relationship

❑Example: The ERM's 1:1 relationship between PROFESSOR and DEPARTMENT

- one department chair—a professor—can chair only one department, and one department can have only one dep



1-1 Relationship

- The implemented 1:1 relationship between PROFESSOR and DEPARTMENT in relational database

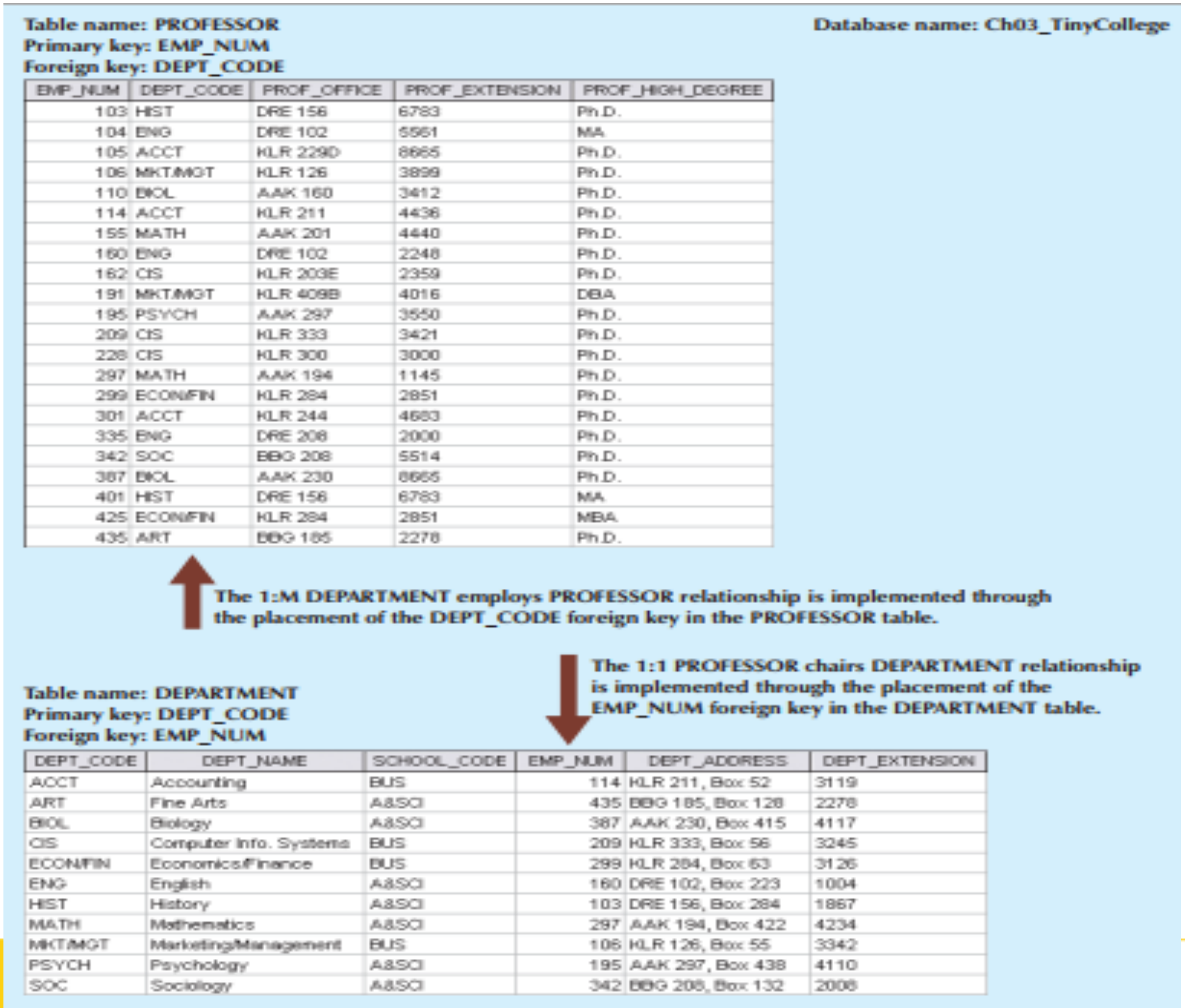


Table name: PROFESSOR
Primary key: EMP_NUM
Foreign key: DEPT_CODE

Database name: Ch03_TinyCollege

EMP_NUM	DEPT_CODE	PROF_OFFICE	PROF_EXTENSION	PROF_HIGH_DEGREE
103	HIST	DRE 156	6783	Ph.D.
104	ENG	DRE 102	5561	MA
105	ACCT	KLR 229D	8665	Ph.D.
106	MKT/MGT	KLR 126	3899	Ph.D.
110	BIOL	AAK 160	3412	Ph.D.
114	ACCT	KLR 211	4436	Ph.D.
155	MATH	AAK 201	4440	Ph.D.
160	ENG	DRE 102	2248	Ph.D.
162	CIS	KLR 203E	2359	Ph.D.
191	MKT/MGT	KLR 409B	4016	DBA
195	PSYCH	AAK 297	3550	Ph.D.
209	CIS	KLR 333	3421	Ph.D.
228	CIS	KLR 300	3000	Ph.D.
297	MATH	AAK 194	1145	Ph.D.
299	ECON/FIN	KLR 284	2851	Ph.D.
301	ACCT	KLR 244	4683	Ph.D.
335	ENG	DRE 208	2000	Ph.D.
342	SOC	BBG 208	5514	Ph.D.
387	BIOL	AAK 230	8665	Ph.D.
401	HIST	DRE 156	6783	MA
425	ECON/FIN	KLR 284	2851	MBA
435	ART	BBG 185	2278	Ph.D.

The 1:M DEPARTMENT employs PROFESSOR relationship is implemented through the placement of the DEPT_CODE foreign key in the PROFESSOR table.

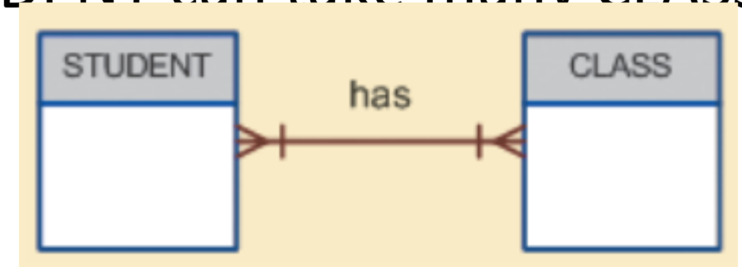
The 1:1 PROFESSOR chairs DEPARTMENT relationship is implemented through the placement of the EMP_NUM foreign key in the DEPARTMENT table.

Table name: DEPARTMENT
Primary key: DEPT_CODE
Foreign key: EMP_NUM

DEPT_CODE	DEPT_NAME	SCHOOL_CODE	EMP_NUM	DEPT_ADDRESS	DEPT_EXTENSION
ACCT	Accounting	BUS	114	KLR 211, Box 52	3119
ART	Fine Arts	A&SCI	435	BBG 185, Box 128	2278
BIOL	Biology	A&SCI	387	AAK 230, Box 415	4117
CIS	Computer Info. Systems	BUS	209	KLR 333, Box 56	3245
ECON/FIN	Economics/Finance	BUS	299	KLR 284, Box 63	3126
ENG	English	A&SCI	160	DRE 102, Box 223	1004
HIST	History	A&SCI	103	DRE 156, Box 284	1867
MATH	Mathematics	A&SCI	297	AAK 194, Box 422	4234
MKT/MGT	Marketing/Management	BUS	106	KLR 126, Box 55	3342
PSYCH	Psychology	A&SCI	195	AAK 297, Box 438	4110
SOC	Sociology	A&SCI	342	BBG 208, Box 132	2008

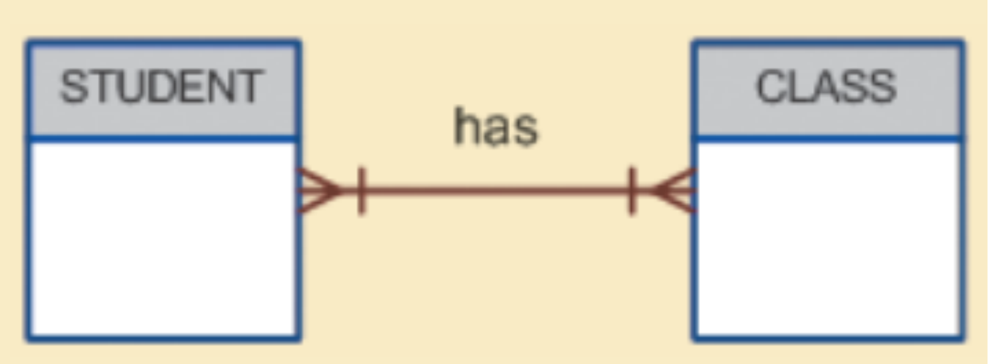
M-N relationship

- The ERM’s M:N relationship between STUDENT and CLASS
  - Each CLASS can have many STUDENTs, and each STUDENT can take many CLASSes.



M-N relationship

- ❑ The ERM’s M:N relationship between STUDENT and CLASS
  - Each CLASS can have many STUDENTs, and each STUDENT can take many CLASSes.



- ❑In relational database, many-to-many (M:N) relationship can easily be implemented by creating a **composite entity** (also referred to as a bridge entity or an associative entity).

M-N relationship

- ❑ The composite entity structure includes—as foreign keys—*at least* the primary keys of the tables that are to be linked.
  - ▪ The database designer has two main options when defining a composite table's primary key:
    - use the combination of those foreign keys or
    - create a new primary key
- ❑ The composite entity is called a **linking table** when implementing a composite entity in relational database.

M-N relationship

❑Converting the M:N relationship into two 1:M relationships

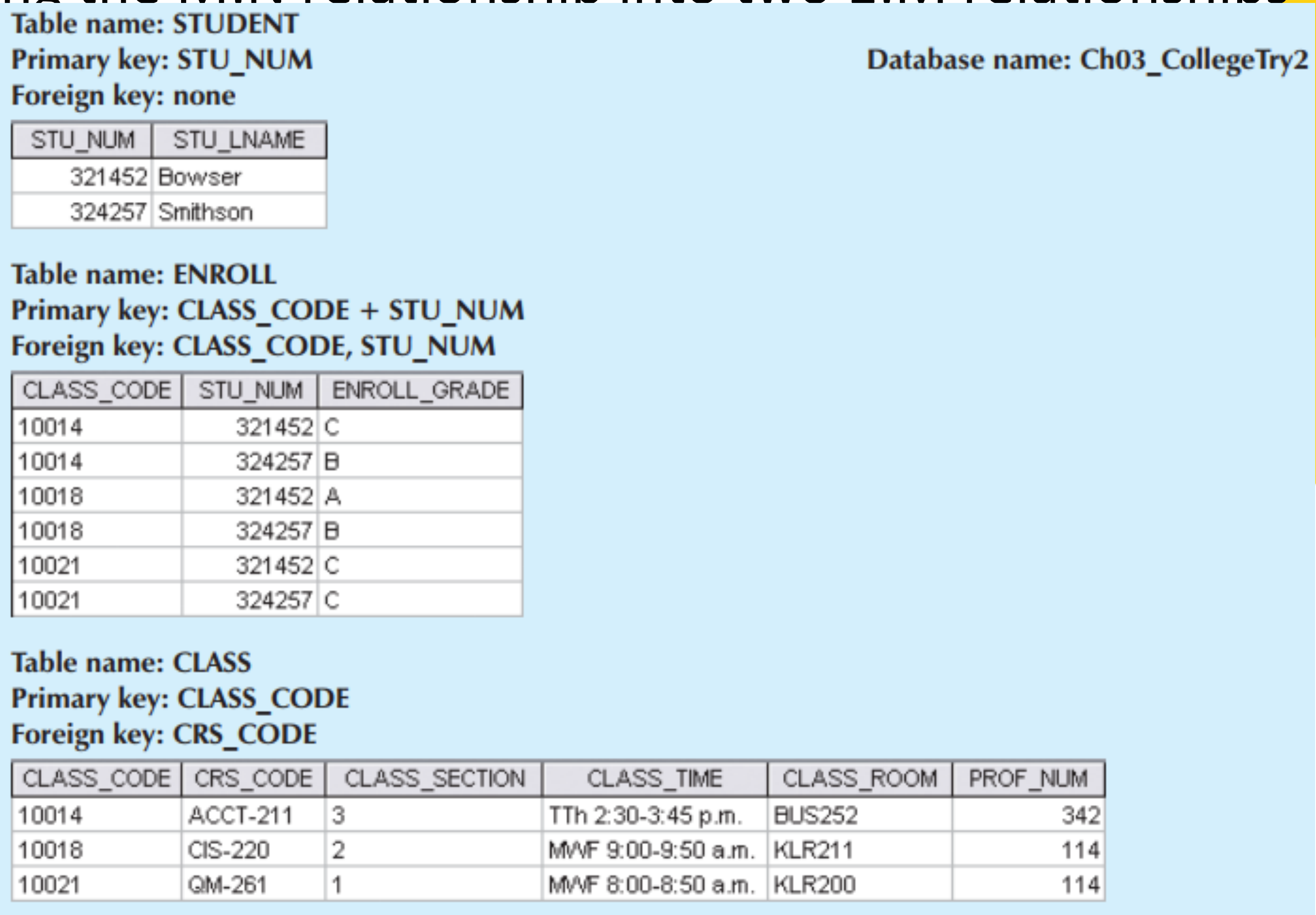


Table name: STUDENT
Primary key: STU_NUM
Foreign key: none

Database name: Ch03_CollegeTry2

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

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

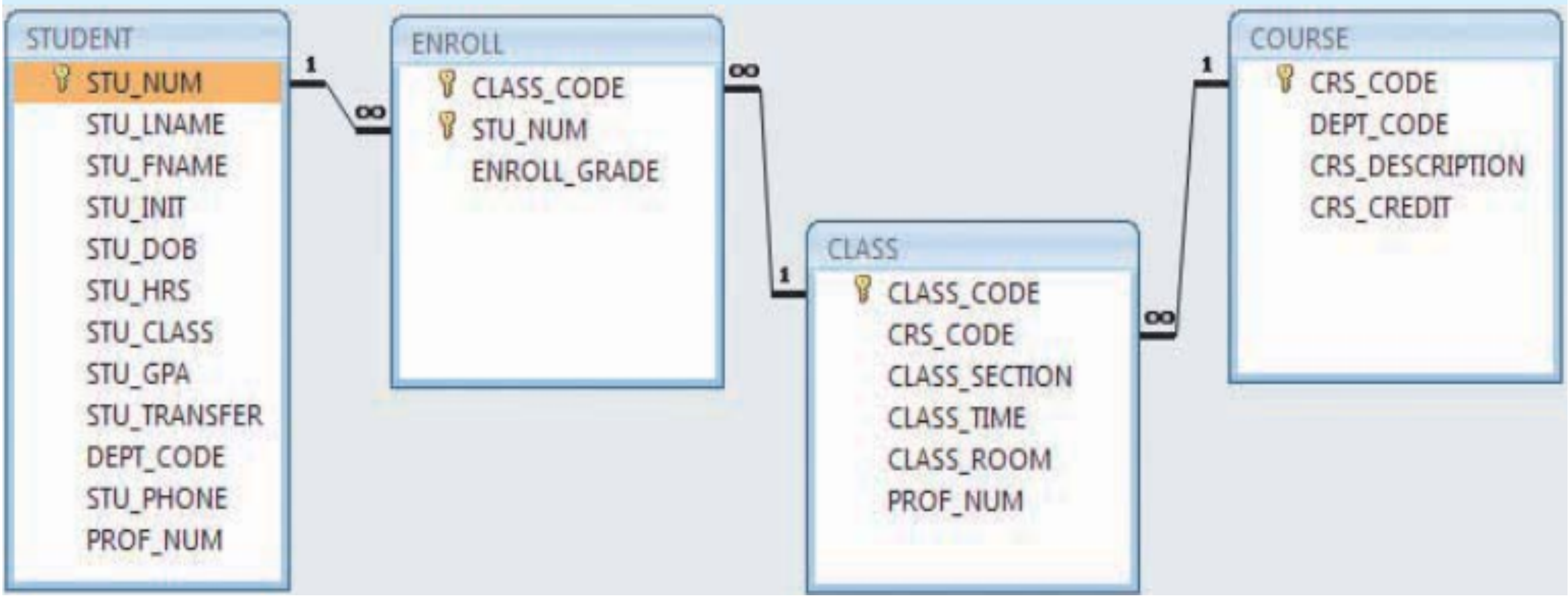
CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Table name: CLASS
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

Relationships within the Relational Database

- The relational diagram for the Ch03_TinyCollege database



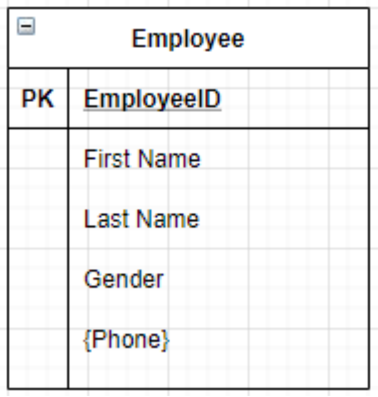
ER Model to Relational mapping

❑Mapping ERM (Entity Relationship Model) to Relation

- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relationship Types
- Step 4: Mapping of Binary 1:N Relationship Types
- Step 5: Mapping of Binary M:N Relationship Types
- Step 6: Mapping Unary Relationship Types
- Step 7: Mapping of Multivalued attributes
- Step 8: Mapping of n-ary Relationship Types
- Step 9: Mapping Supertype/Subtype Relationships

Step 1: Mapping strong/regular Entity

- For each strong/regular entity, create a relation that includes all the simple attributes
- Primary key of the entity becomes the primary key of the relation.
- Exclude multivalued attribute from the mapping relation.
- Example:



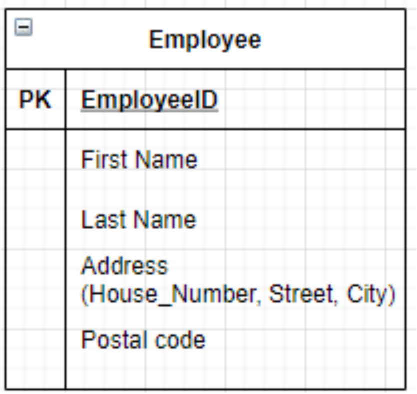
Employee	
PK	EmployeeID
	First Name
	Last Name
	Gender
	{Phone}

⇒Employee(EmployeeID, FirstName, LastName, Gender)

Note: Phone is a multivalued attribute.

Step 1: Mapping strong/regular Entity

- When a regular entity type has a **composite attribute**, only the simple components of the composite attribute are included in the new relation as its attributes
- Example:

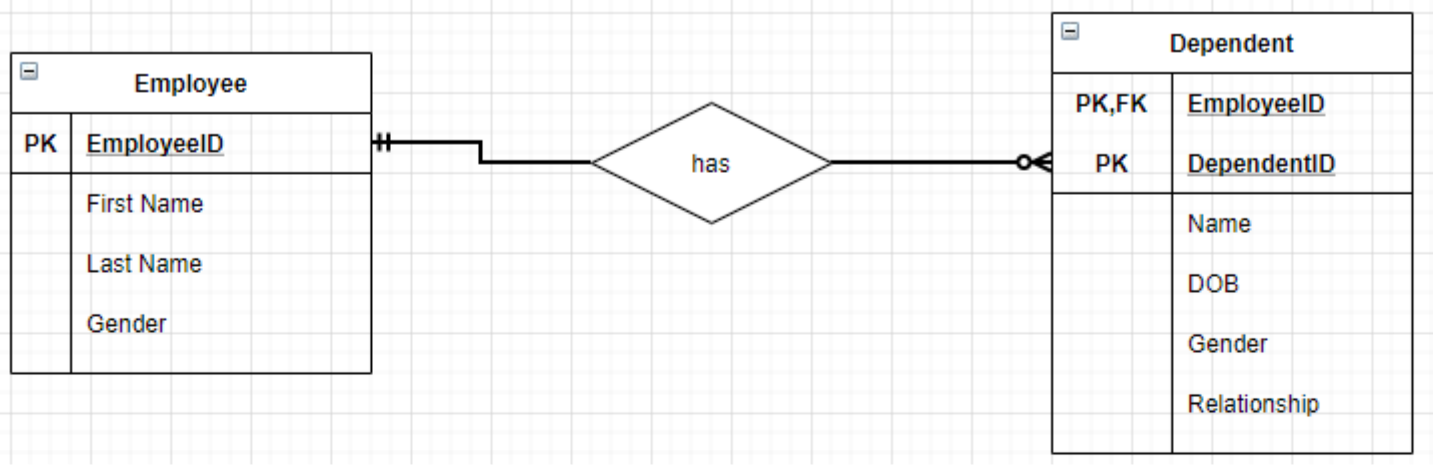


	Employee
PK	EmployeeID
	First Name
	Last Name
	Address (House_Number, Street, City)
	Postal code

⇒Employee(EmployeeID, FirstName, LastName, House_Number, Street, City, PostalCode)

Step 2: Mapping of Weak Entity

- Create separate relation and include all simple attributes
- The primary key of the relation is the combination of all the primary key attributes from the owner and the partial key of the weak entity, if any.



Employee	
PK	EmployeeID
	First Name
	Last Name
	Gender

has

Dependent	
PK,FK	EmployeeID
PK	DependentID
	Name
	DOB
	Gender
	Relationship

=> Dependent(EmployeeID, DependentID, Name, DOB, Gender, Relationship)

Employee(EmployeeID, FirstName, LastName, Gender)

Step 3: Mapping Binary 1:1 relationship

❑**Step 3: Mapping Binary 1:1 relationship**

Before tackling a 1:1 relationship, we need to know its **optionality**.

There are three possibilities the relationship can be:

1. mandatory at both ends
2. mandatory at one end and optional at the other
3. optional at both ends

Step 3: Mapping Binary 1:1 relationship

1. Mandatory at both ends

❑**Combine two entity into one when the relationship is mandatory at both ends.**

- The choice of which entity type subsumes the other depends on which is the *most important entity type* (more attributes, better key, semantic nature of them).
- The *key of the subsumed entity type becomes a normal attribute*

❑**When not to combine a 1:1 mandatory relationship:**

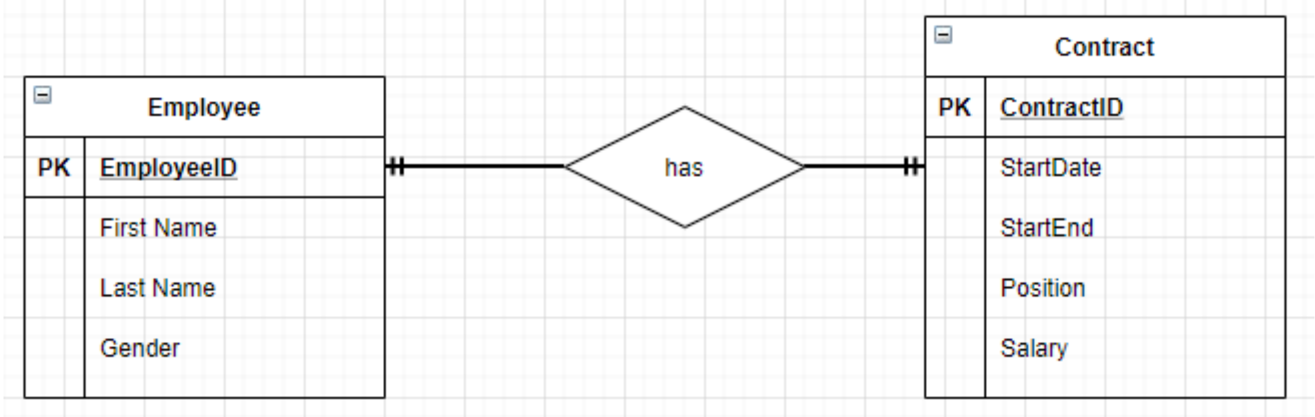
- the two entity types represent different entities in the ‘real world’.
- the entities participate in very different relationships with other entities.
- efficiency considerations when fast responses are required or different patterns of updating occur to the two different entity types

=> The primary key of one entity type comes the foreign key in the other.

Step 3: Mapping Binary 1:1 relationship

❑**Example:** Two entity types; employee and contract.

- Each member of employee must have one contract and each contract must have one member of employee associated with it.
- It is therefore a mandatory relations at both ends.



⊟	Employee
PK	EmployeeID
	First Name
	Last Name
	Gender

⊟	Contract
PK	ContractID
	StartDate
	StartEnd
	Position
	Salary

Step 3: Mapping Binary 1:1 relationship

3 options to mapping:

- **Combine two entity into one.**

  Employee(EmployeeID, FirstName, LastName, Gender, ContractID, StartDate, StartEnd, Position, Salary)

❑ **or kept apart and a foreign key used**

  Employee(EmployeeID, FirstName, LastName, Gender, *ContractID*)

  Contract(ContractID, StartDate, StartEnd, Position, Salary)

❑ **or**

  Employee(EmployeeID, FirstName, LastName, Gender)

  Contract(ContractID, StartDate, StartEnd, Position, Salary, *EmployeeID*)

Step 3: Mapping Binary 1:1 relationship

2. Mandatory at one end and optional at the other

- *Take the primary key from the 'mandatory end' and add it to the 'optional end' as a foreign key.*
- The entity type of the optional end may be subsumed into the mandatory end as in the previous example.
- It is better NOT to subsume the mandatory end into the optional end as this will create **null entries**.
- Given entity types A and B, where A, B is in a relationship where the A end it optional, the result would be:

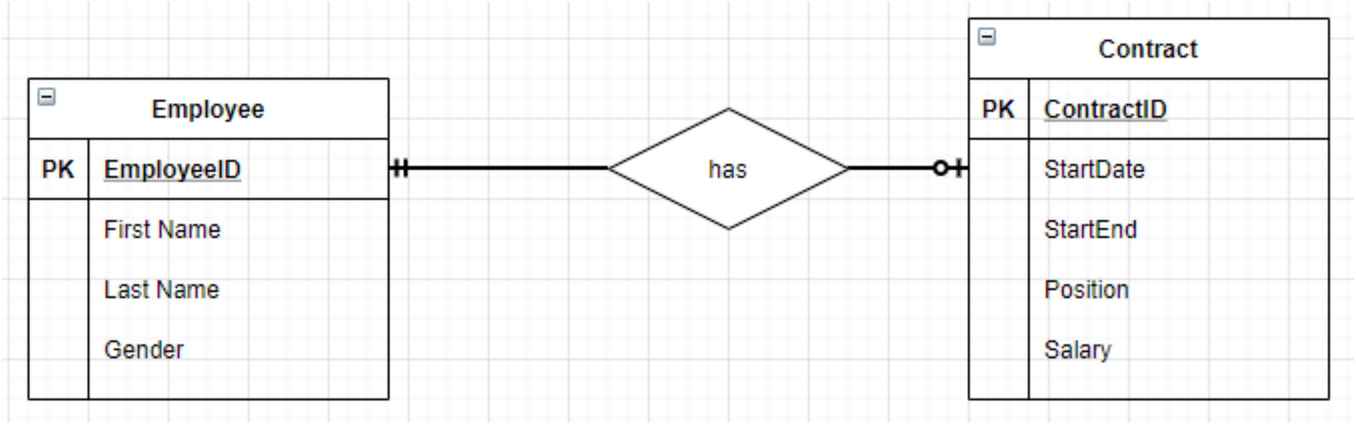
A (<u>primary key</u>,attribute,...,*foreign key to B*)

B (<u>primary key</u>,attribute,...)

Step 3: Mapping Binary 1:1 relationship

2. Mandatory at one end and optional at the other

- Example: Contract is optional (each employee may have at most one contract)



⊟	Employee
PK	EmployeeID
	First Name
	Last Name
	Gender

⊟	Contract
PK	ContractID
	StartDate
	StartEnd
	Position
	Salary

=> Employee(<u>EmployeeID</u>, FirstName, LastName, Gender)

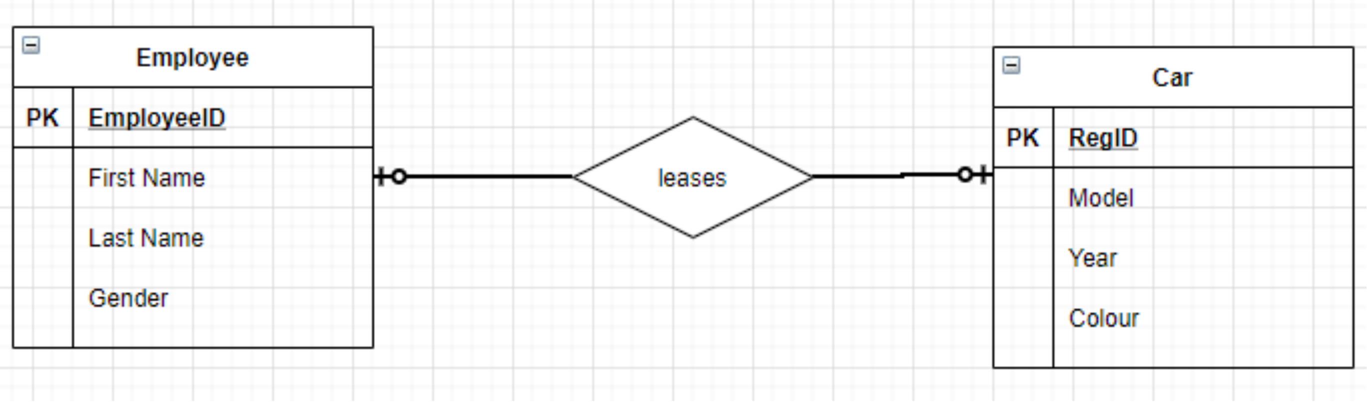
=> Contract(<u>ContractID</u>, StartDate, StartEnd, Position, Salary, *EmployeeID*)

Step 3: Mapping Binary 1:1 relationship

2. Optional at both ends

❑Use a foreign key approach.

❑Example: Each staff member may lease up to one car, Each car may be leased by at most one member of staff



Employee	
PK	EmployeeID
	First Name
	Last Name
	Gender

Car	
PK	RegID
	Model
	Year
	Colour

=>Can not combine two entity into one.

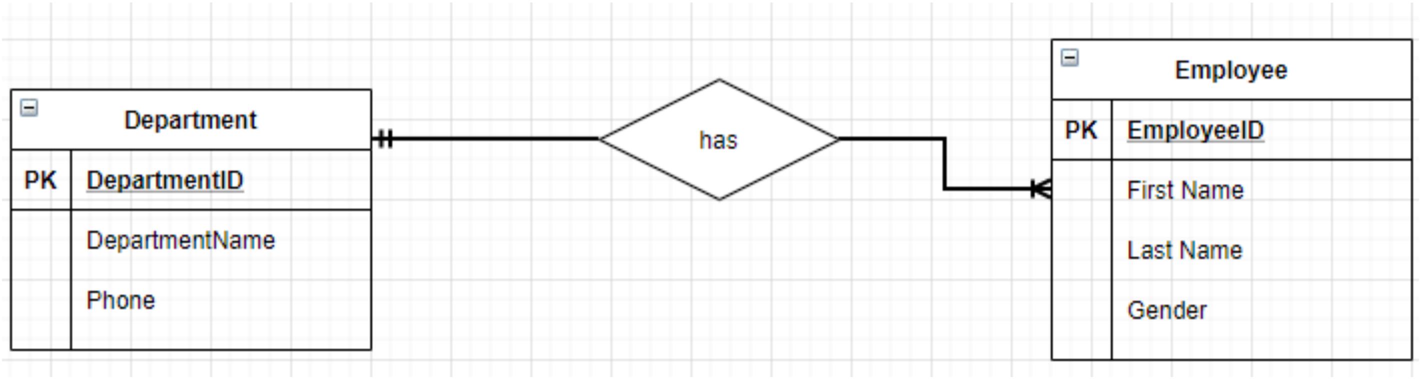
Employee(EmployeeID, FirstName, LastName, Gender)

Car(RegID, Model, Year, Colour, *EmployeeID*)

Step 4: Mapping Binary 1:M relationship

❑The primary key on the ‘one side’ of the relationship is added to the ‘many side’ as a foreign key.

❑Example:



Department

PK DepartmentID

DepartmentName

Phone

has

Employee

PK EmployeeID

First Name

Last Name

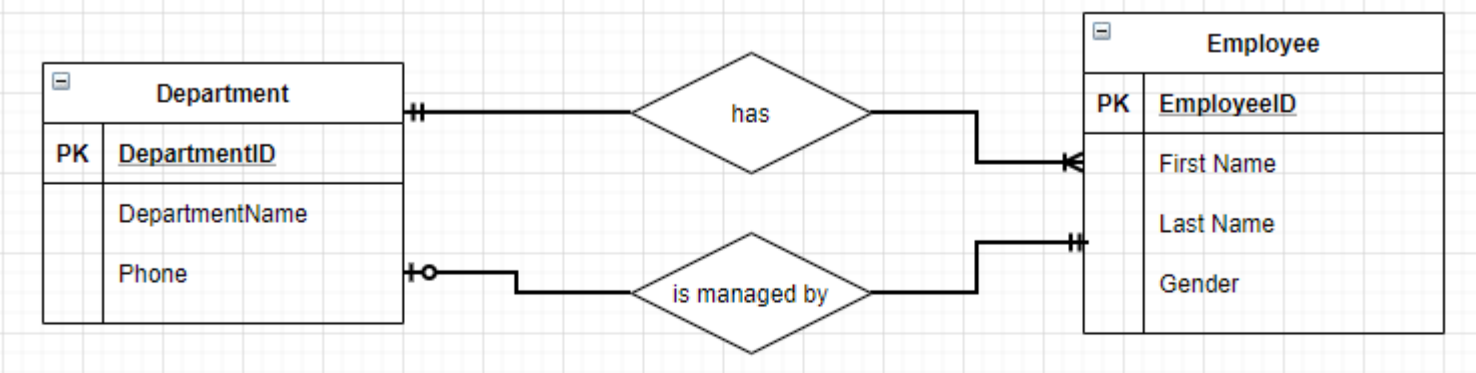
Gender

=> Department(DepartmentID, DempartmentName, Phone)

=> Employee(EmployeeID, FirstName, LastName, Gender, *DepartmentID*)

Step 4: Mapping Binary 1:M relationship

- **Parallel relationships** occur when there are two or more relationships between two entity types
- Example:



- In order to distinguish between the two roles we can give the foreign keys different names.

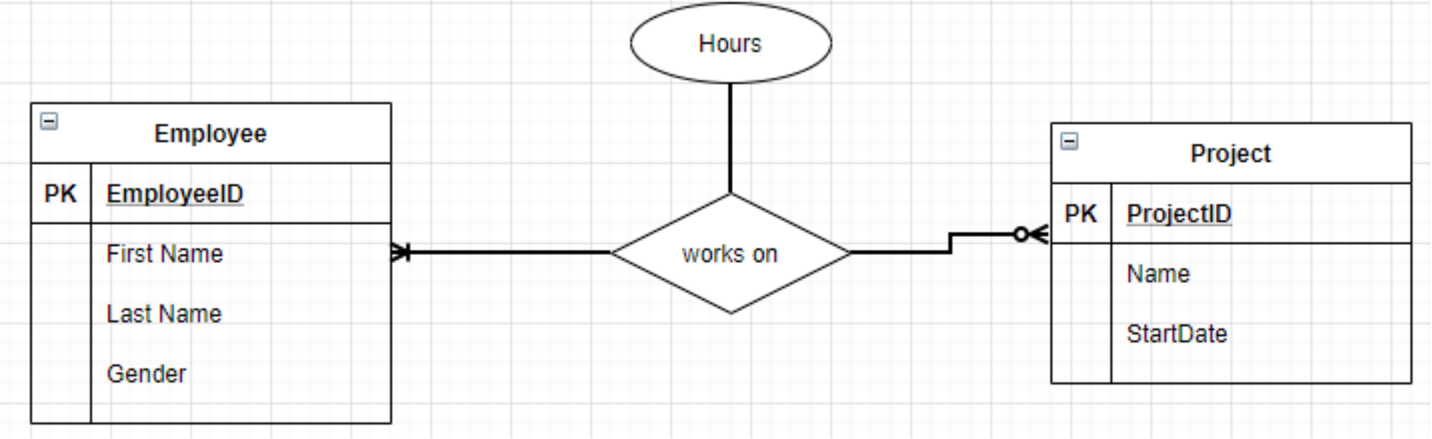
=>Employee(<u>EmployeeID</u>, FirstName, LastName, Gender, *DepartmentID*)

=>Department(<u>DepartmentID</u>, DepartmentName, *ManagerID*)

Step 5: Mapping Binary M:N relationship

❑**Create a new relation** containing the primary keys of both participating entity types and **descriptive attribute** (if any)

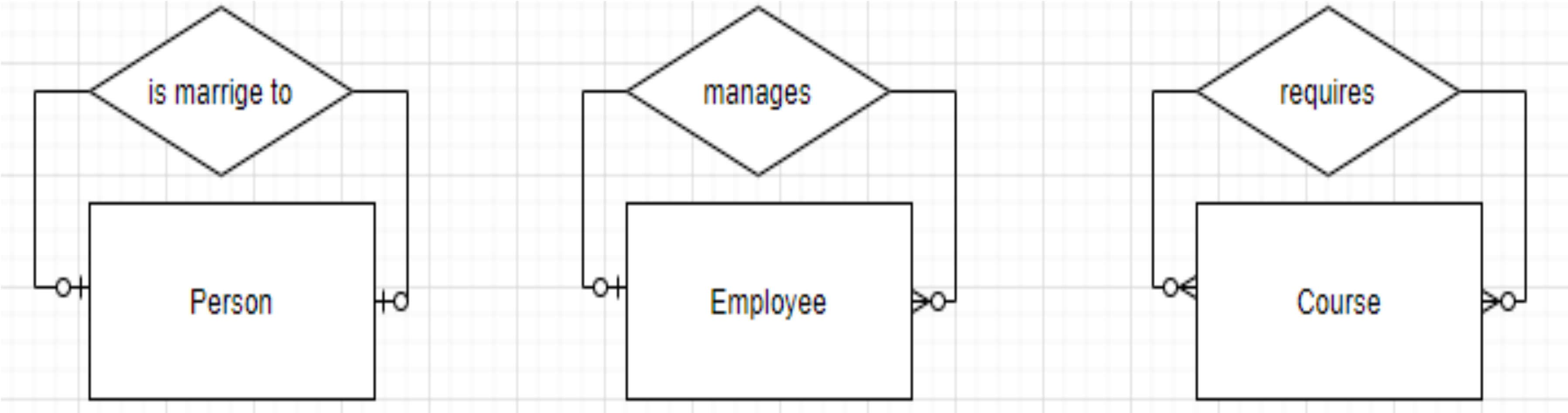
❑These primary keys form a **composite primary key** of the new relation.



=> WorksOn(<u>EmployeeID</u>, <u>ProjectID</u>, Hours)

Step 6: Mapping Unary relationship

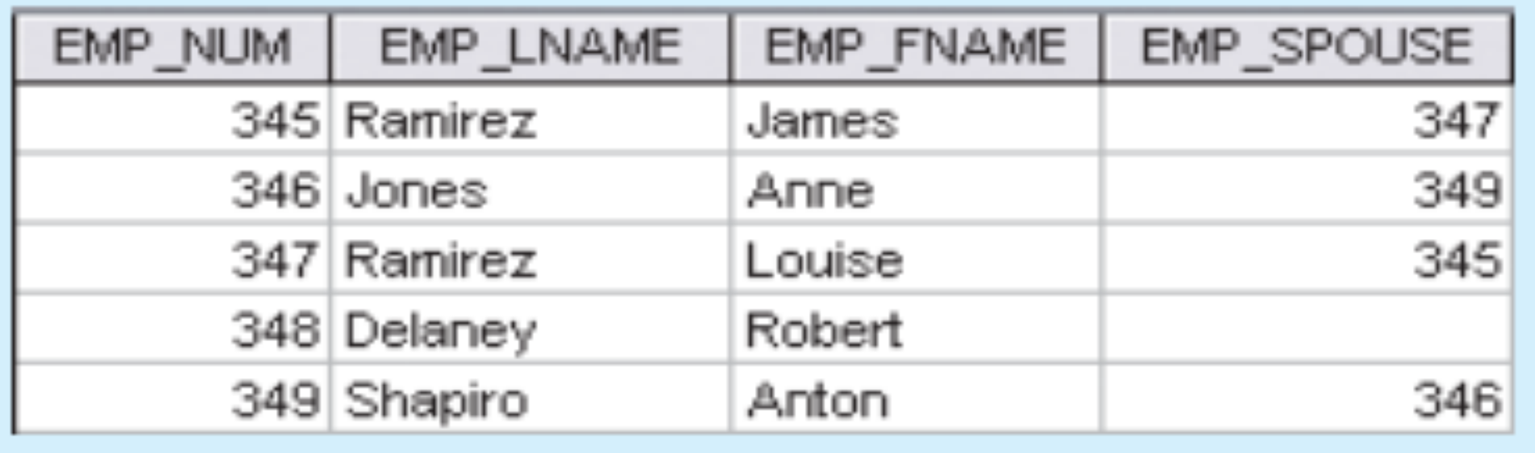
- ❑Unary relationships (or recursive relationships) is one in which a relationship can exist between occurrences of the same entity set.
- ❑Example:



1-1 relationship **1-M relationship** **M-N relationship**

Step 6: Mapping Unary relationship

- ❑Mapping the 1:1 unary relationship “EMPLOYEE is married to EMPLOYEE”
  => Employee(EmpNum, Emp_Lname, Emp_Fname, *Emp_Spouse*)
- ❑Implementation



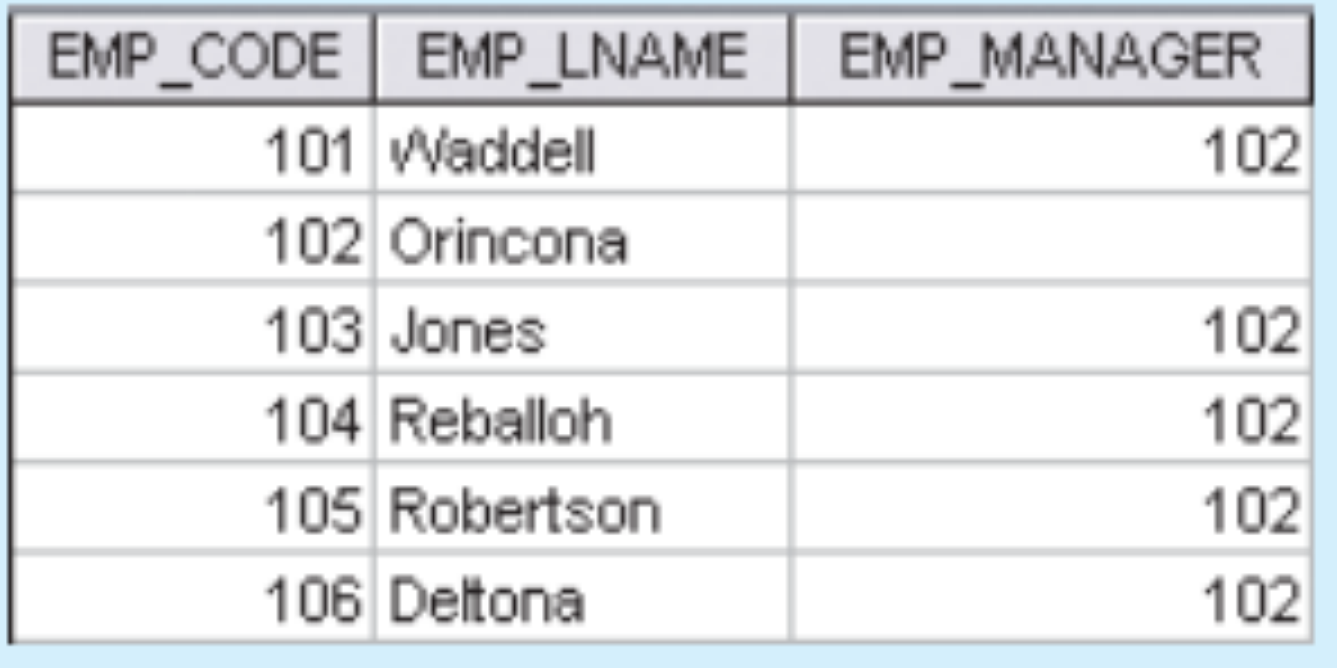
EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_SPOUSE
345	Ramirez	James	347
346	Jones	Anne	349
347	Ramirez	Louise	345
348	Delaney	Robert	
349	Shapiro	Anton	346

=>The foreign key is the primary key of the same table, but is given a different name.

Step 6: Mapping Unary relationship

❑Mapping the 1:M unary relationship “**EMPLOYEE manages EMPLOYEE**”
=> Employee(Emp_Code, Emp_Lname, *Emp_Manager*)

❑Implementation



EMP_CODE	EMP_LNAME	EMP_MANAGER
101	Waddell	102
102	Orincona	
103	Jones	102
104	Reballoh	102
105	Robertson	102
106	Deltona	102

=>The foreign key is the primary key of the same table, but is given a different name.

Step 6: Mapping Unary relationship

- ❑Mapping the M-N unary relationship "**COURSE requires COURSE**"

=> Course(Crs_code, Dept_code, Crs_Description, Crs_credit)

=> PreReq(Crs_code, Pre_Take)

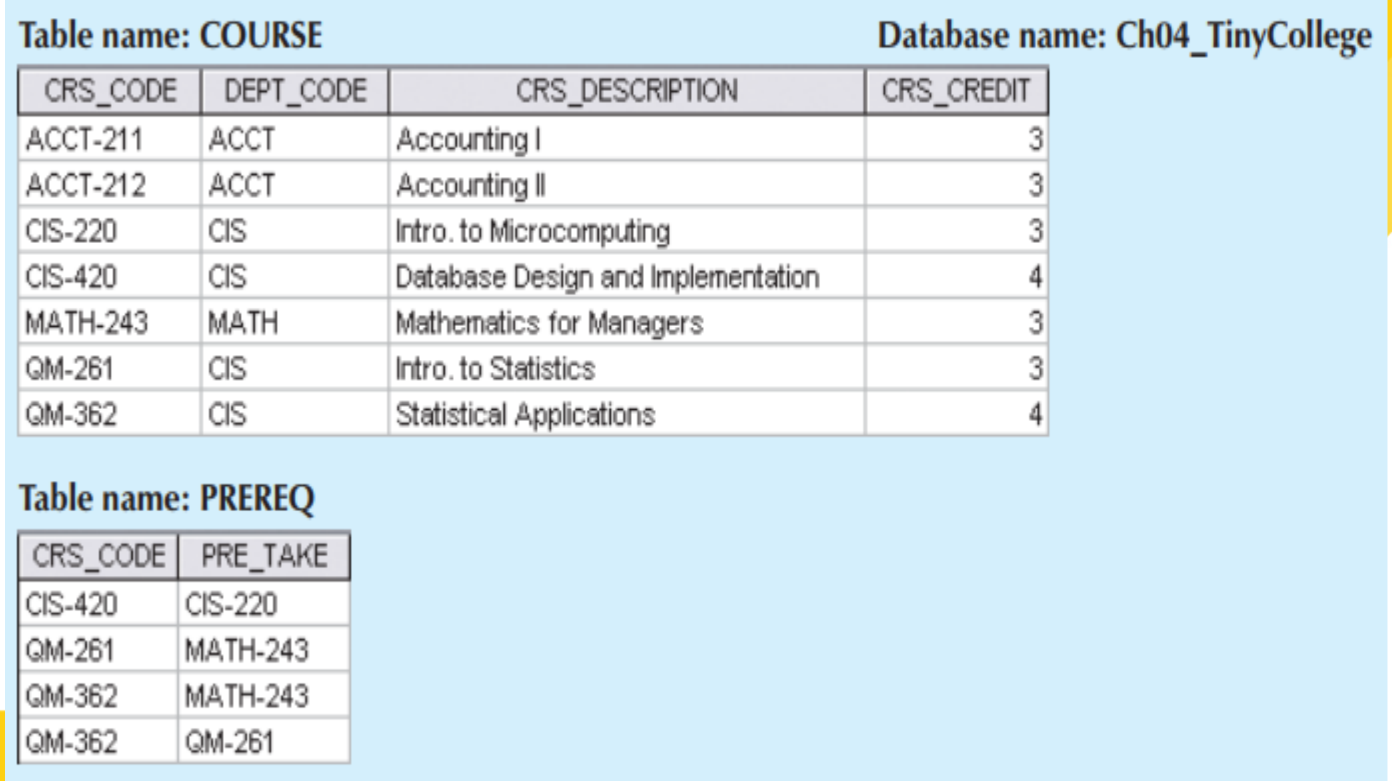


Table name: COURSE

Database name: Ch04_TinyCollege

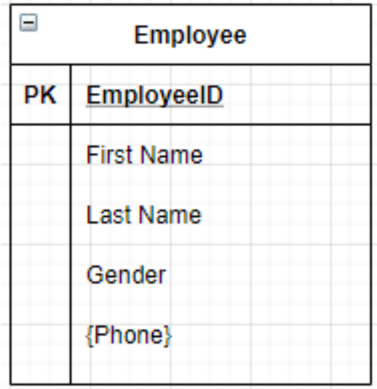
CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: PREREQ

CRS_CODE	PRE_TAKE
CIS-420	CIS-220
QM-261	MATH-243
QM-362	MATH-243
QM-362	QM-261

Step 7: Mapping Multivalued Attributes

- ❑Create a new relation R for each multivalued attribute
- ❑Add primary key of the original entity to the new relation R as a foreign key.
- ❑The primary key of R is the combination of R and the original entity. If the multivalued attribute is composite, we include its simple components.



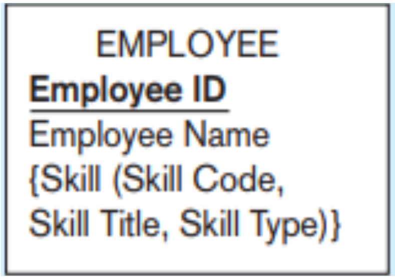
⊟	Employee
PK	EmployeeID
	First Name
	Last Name
	Gender
	{Phone}

=> Phone(EmployeeID, PhoneNumber, type)

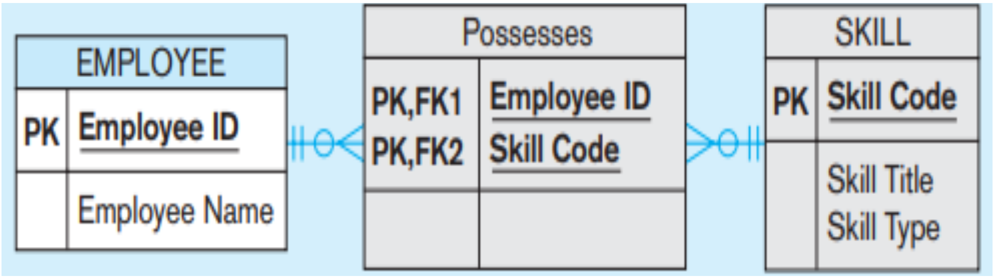
=>Employee(EmployeeID, FirstName, LastName, Gender)

Step 7: Mapping Multivalued Attributes

❑ Another example: In ER model



EMPLOYEE
Employee ID
Employee Name
{Skill (Skill Code,
Skill Title, Skill Type)}



EMPLOYEE

PK	Employee ID
	Employee Name

Possesses

PK,FK1	Employee ID
PK,FK2	Skill Code

SKILL

PK	Skill Code
	Skill Title Skill Type

❑In the relation shema:

=>Skill(SkillCode, Skill Title, Skill Type)

=>Possesses(EmployeeID, SkillCode)

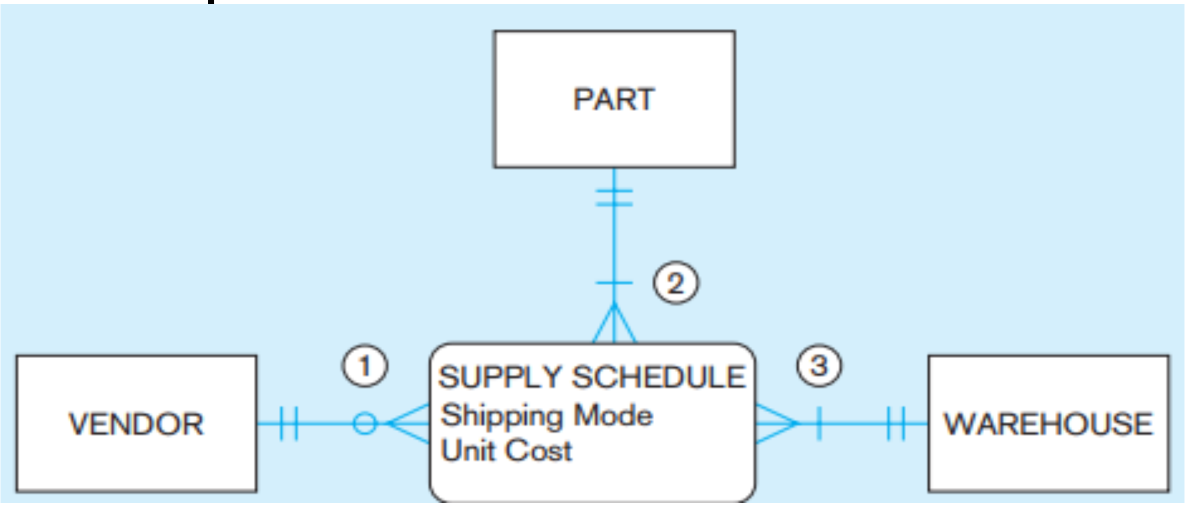
=>Employee(EmployeeID, EmployeeName)

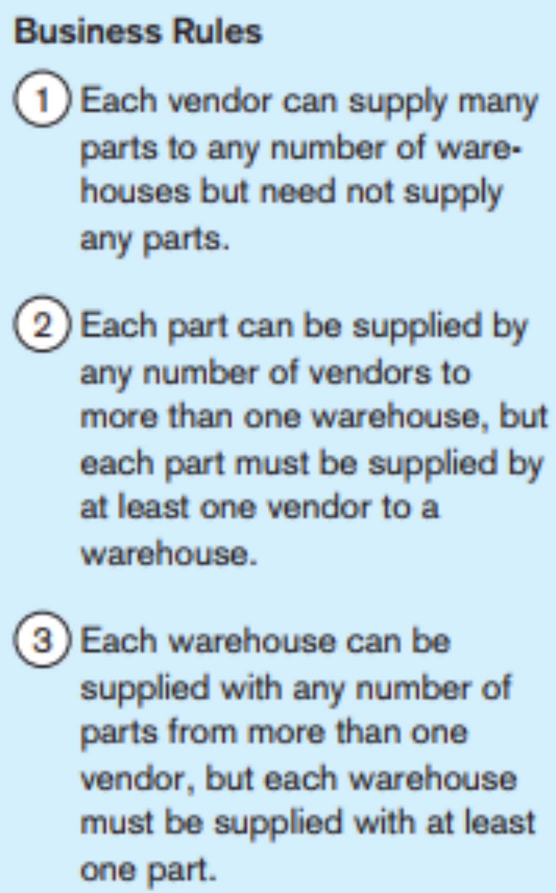
Step 8: Mapping n-ary Relationship types

- Mapping Ternary (and n-ary) relationship type:
  - One relation for each entity and one for the associative entity
  - Associative entity has foreign keys to each entity in the relationship

Step 8: Mapping n-nary Relationship types

❑Example





Business Rules

1 Each vendor can supply many parts to any number of warehouses but need not supply any parts.

2 Each part can be supplied by any number of vendors to more than one warehouse, but each part must be supplied by at least one vendor to a warehouse.

3 Each warehouse can be supplied with any number of parts from more than one vendor, but each warehouse must be supplied with at least one part.

❑In relational schema

=>Vendor(VendorID,…)

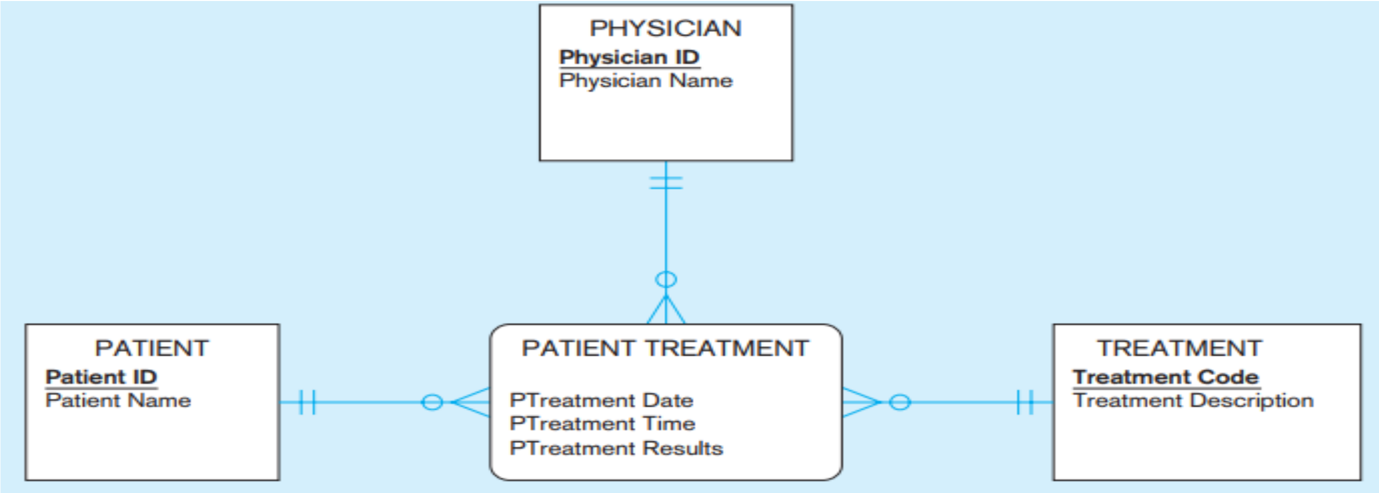
=>Warehouse(WareHouseID,…)

=>Part(PartID,…)

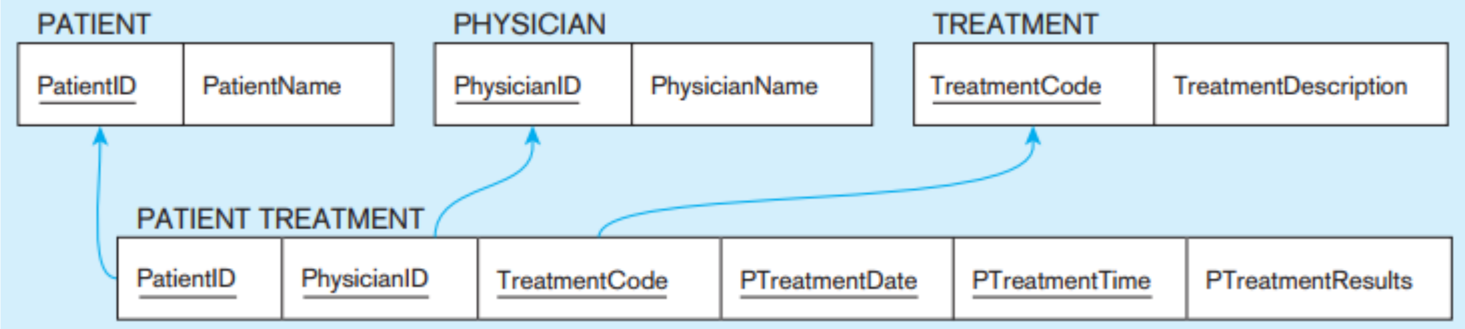
=>Supply(VendorID, WareHouseID, PartID, shippingMode, UnitCode)

Step 8: Mapping n-nary Relationship types

- ❑Another example:



- ❑In relational schema

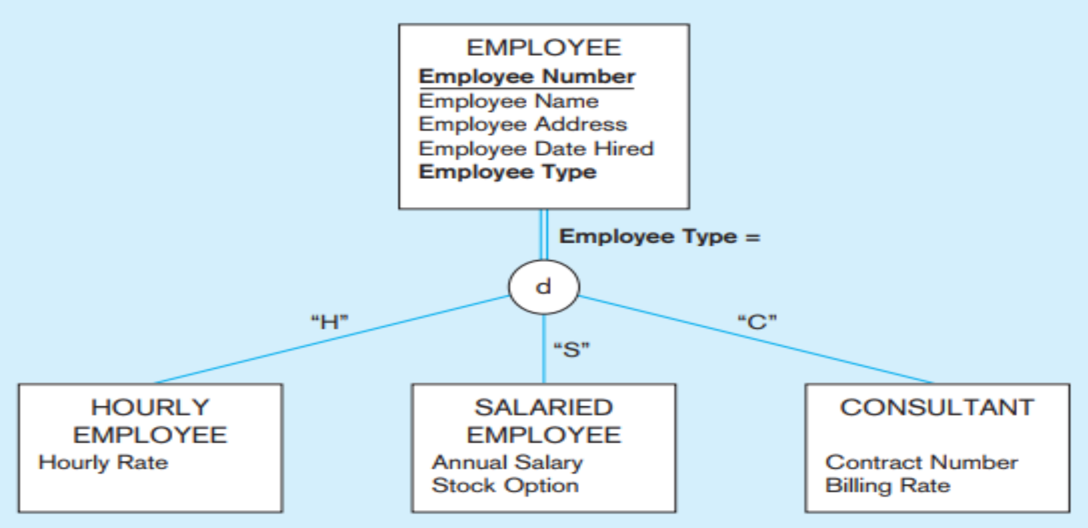


Mapping Supertype/Subtype Relationships

- ❑One relation for supertype and for each subtype
- ❑Supertype attributes (including identifier and subtype discriminator) go into supertype relation
- ❑Subtype attributes go into each subtype; primary key of supertype relation also becomes primary key of subtype relation
- ❑1:1 relationship established between supertype and each subtype, with supertype as primary table

Mapping Supertype/Subtype Relationships

- ❑Example



EMPLOYEE
Employee Number
Employee Name
Employee Address
Employee Date Hired
Employee Type

Employee Type =

d

"H"

"S"

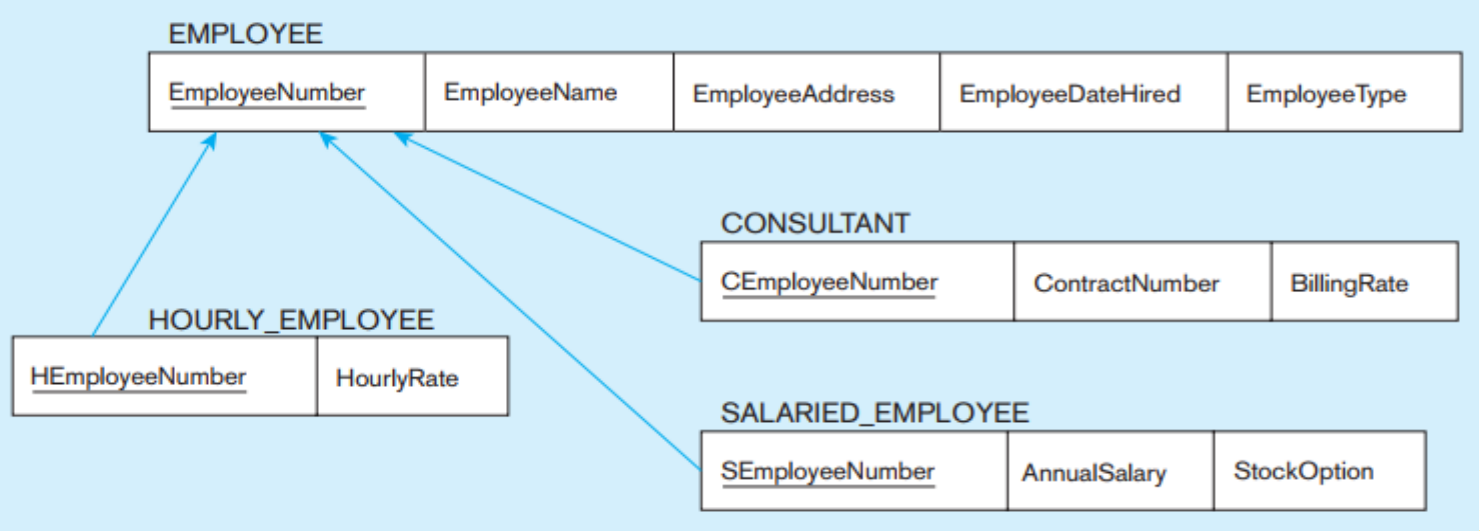
"C"

HOURLY EMPLOYEE
Hourly Rate

SALARIED EMPLOYEE
Annual Salary
Stock Option

CONSULTANT
Contract Number
Billing Rate

- ❑Mapping



EMPLOYEE

EmployeeNumber	EmployeeName	EmployeeAddress	EmployeeDateHired	EmployeeType

CONSULTANT

CEmployeeNumber	ContractNumber	BillingRate

HOURLY_EMPLOYEE

HEmployeeNumber	HourlyRate

SALARIED_EMPLOYEE

SEmployeeNumber	AnnualSalary	StockOption

Thank You !