

# **Unit 3: The Relational Database Model**

(Part 2)

# Learning Objectives

- In this chapter, you will learn:
  - About relational database operators, the data dictionary, and the system catalog
  - How data redundancy is handled in the relational database model
  - Why indexing is important



# Relational Algebra

- Theoretical way of manipulating table contents using relational operators
- A set of mathematical principles that form the basis for manipulating relational table contents;
- The eight main functions are SELECT, PROJECT, JOIN, INTERSECT, UNION, DIFFERENCE, PRODUCT, and DIVIDE.



# Relational Algebra

- **Relvar:** (Relational Variable) Variable that holds a relation
  - Heading contains the names of the attributes and
  - the body contains the relation
- Relational operators have the property of closure
  - **Closure:** Use of relational algebra operators on existing relations produces new relations

# Relational Set Operators

## Select (Restrict)

- Unary operator that yields a horizontal subset of a table

## Project

- Unary operator that yields a vertical subset of a table

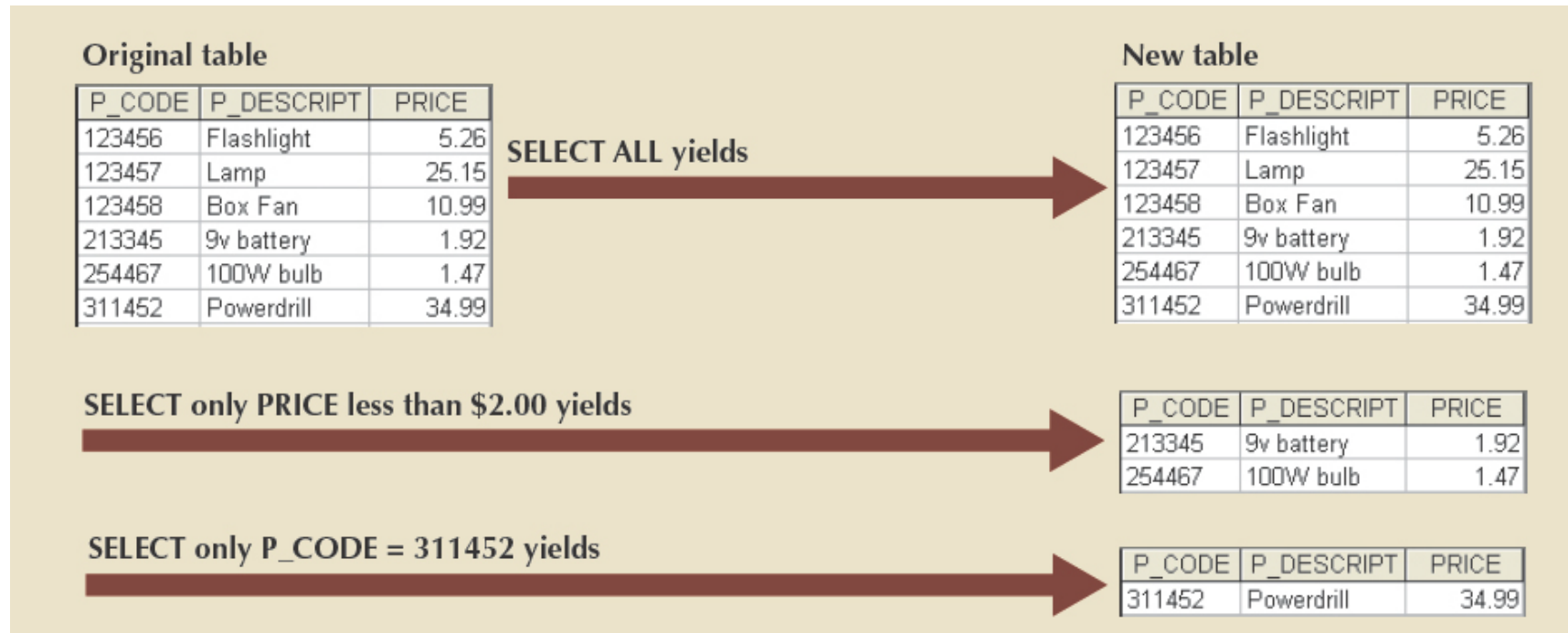
## Union

- Combines all rows from two tables, excluding duplicate rows
- **Union-compatible:** Tables share the same number of columns, and their corresponding columns share compatible domains

## Intersect

- Yields only the rows that appear in both tables
- Tables must be union-compatible to yield valid results

# Figure 3.4 - Select



# Relational Set Operators

## Select (Restrict)

- Unary operator that yields a horizontal subset of a table

## Project

- Unary operator that yields a vertical subset of a table

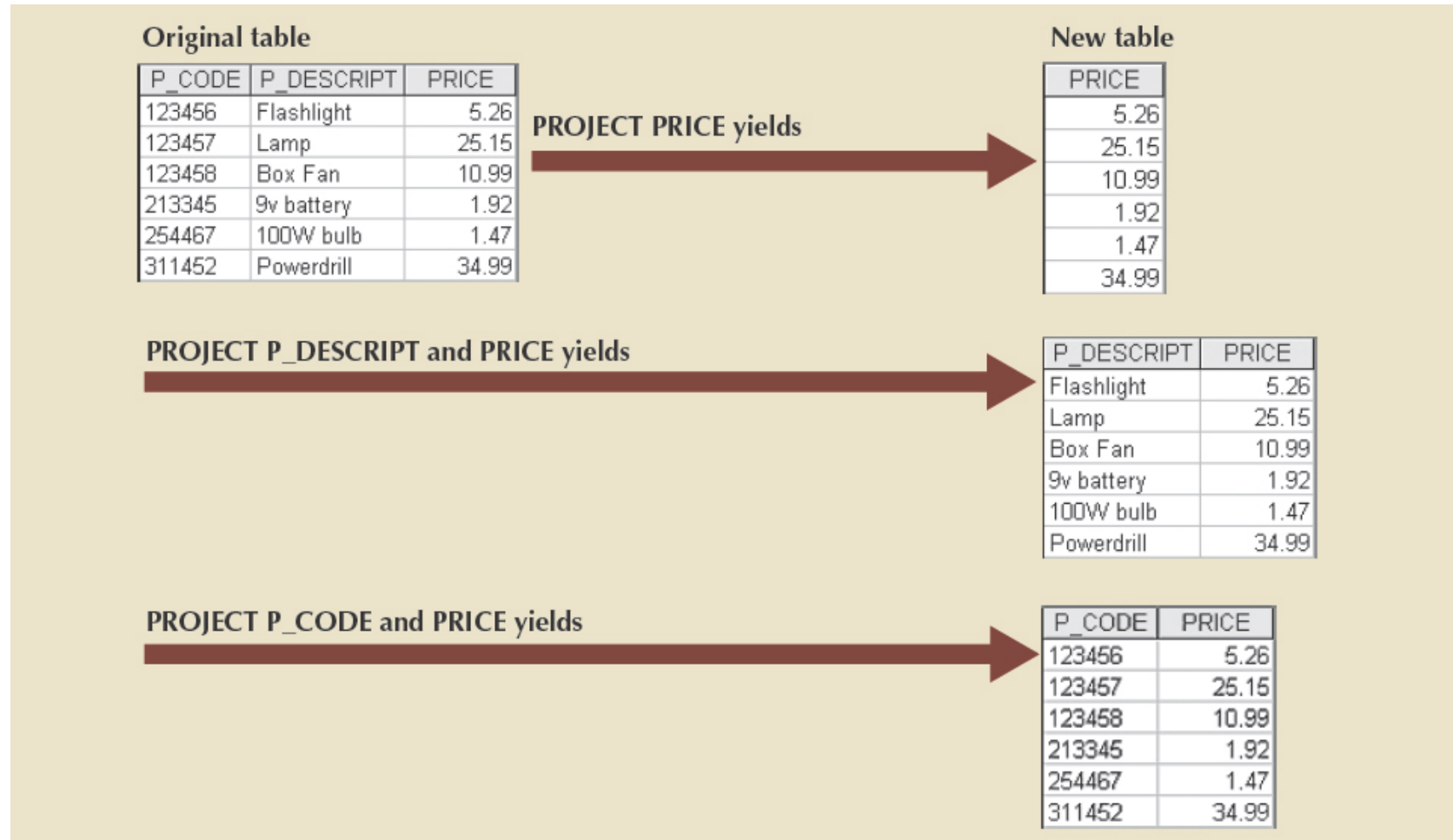
## Union

- Combines all rows from two tables, excluding duplicate rows
- **Union-compatible:** Tables share the same number of columns, and their corresponding columns share compatible domains

## Intersect

- Yields only the rows that appear in both tables
- Tables must be union-compatible to yield valid results

# Figure 3.5 - Project





# Relational Set Operators

## Select (Restrict)

- Unary operator that yields a horizontal subset of a table

## Project

- Unary operator that yields a vertical subset of a table

## Union



- Combines all rows from two tables, excluding duplicate rows
- **Union-compatible:** Tables share the same number of columns, and their corresponding columns share compatible domains

## Intersect


- Yields only the rows that appear in both tables
- Tables must be union-compatible to yield valid results

# Figure 3.6 - Union and Figure 3.7 - Intersect

## Figure 3.6 - Union

P_CODE	P_DESCRIPT	PRICE	UNION			yields	P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26					123456	Flashlight	5.26
123457	Lamp	25.15					123457	Lamp	25.15
123458	Box Fan	10.99					123458	Box Fan	10.99
213345	9v battery	1.92					213345	9v battery	1.92
254467	100W bulb	1.47					254467	100W bulb	1.47
311452	Powerdrill	34.99					311452	Powerdrill	34.99
345678	Microwave	160.00					345678	Microwave	160.00
345679	Dishwasher	500.00					345679	Dishwasher	500.00
123458	Box Fan	10.99					123458	Box Fan	10.99

## Figure 3.7 - Intersect

STU_FNAME	STU_LNAME	INTERSECT		yields	STU_FNAME	STU_LNAME
George	Jones				Franklin	Lopez
Jane	Smith				William	Turner
Peter	Robinson				Franklin	Johnson
Franklin	Johnson				Susan	Rogers
Martin	Lopez					



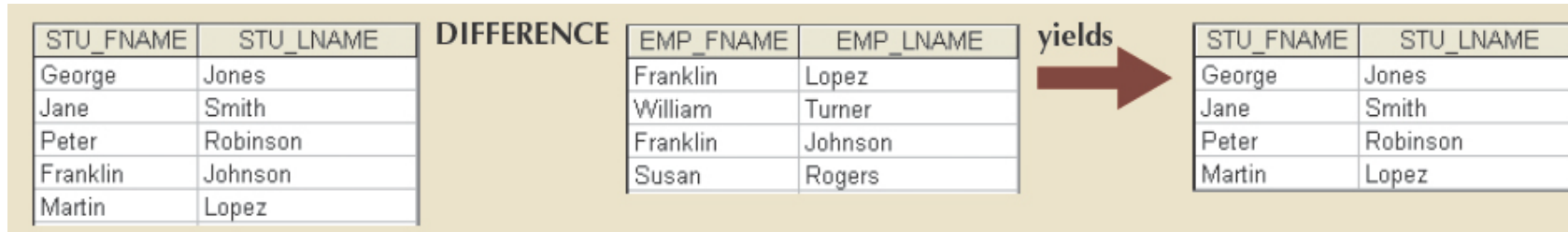
# Relational Set Operators (1 of 2)

- **Difference**

- Yields all rows in one table that are not found in the other table
- Tables must be union-compatible to yield valid results
- However, note that subtracting the first table from the second table is not the same as subtracting the second table from the first table.

# Figure 3.8 – Difference

Figure 3.8 – Difference





# Relational Set Operators (1 of 2)

- **Product**
  - Yields all possible pairs of rows from two tables
  - Also known as the Cartesian product

# Figure 3.9 - Product

Figure 3.9 - Product

P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26
123457	Lamp	25.15
123458	Box Fan	10.99
213345	9v battery	1.92
254467	100W bulb	1.47
311452	Powerdrill	34.99

PRODUCT

STORE	aisle	shelf
23	W	5
24	K	9
25	Z	6

yields



P_CODE	P_DESCRIPT	PRICE	STORE	aisle	shelf
123456	Flashlight	5.26	23	W	5
123456	Flashlight	5.26	24	K	9
123456	Flashlight	5.26	25	Z	6
123457	Lamp	25.15	23	W	5
123457	Lamp	25.15	24	K	9
123457	Lamp	25.15	25	Z	6
123458	Box Fan	10.99	23	W	5
123458	Box Fan	10.99	24	K	9
123458	Box Fan	10.99	25	Z	6
213345	9v battery	1.92	23	W	5
213345	9v battery	1.92	24	K	9
213345	9v battery	1.92	25	Z	6
311452	Powerdrill	34.99	23	W	5
311452	Powerdrill	34.99	24	K	9
311452	Powerdrill	34.99	25	Z	6
254467	100W bulb	1.47	23	W	5
254467	100W bulb	1.47	24	K	9
254467	100W bulb	1.47	25	Z	6

# Relational Set Operators (2 of 2)

- **Join**

- Allows information to be intelligently combined from two or more tables

## Figure 3.10 - Two Tables That Will Be Used in JOIN Illustrations

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445



# Types of Joins (1 of 2)

- **Natural join:** Links tables by selecting only the rows with common values in their common attributes
  - **Join columns:** Common columns
  - Also simply called - JOIN
- **A natural join is the result of a three-stage process:**
  - First, a PRODUCT of the tables is created
  - Second, a SELECT is performed on the output of Step 1 to yield only the rows for which the join column(s) values are equal.
  - A PROJECT is performed on the results of Step 2 to yield a single copy of each attribute, thereby eliminating duplicate columns.

## Figure 3.10 - Two Tables That Will Be Used in JOIN Illustrations

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

# Exercise: How many rows will a PRODUCT of these two tables generate

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

# FIGURE 3.11 NATURAL JOIN, STEP 1: PRODUCT

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1132445	Walker	32145	231	125	6152439887
1132445	Walker	32145	231	167	6153426778
1132445	Walker	32145	231	231	6152431124
1132445	Walker	32145	231	333	9041234445
1217782	Adares	32145	125	125	6152439887
1217782	Adares	32145	125	167	6153426778
1217782	Adares	32145	125	231	6152431124
1217782	Adares	32145	125	333	9041234445
1312243	Rakowski	34129	167	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1312243	Rakowski	34129	167	231	6152431124
1312243	Rakowski	34129	167	333	9041234445
1321242	Rodriguez	37134	125	125	6152439887
1321242	Rodriguez	37134	125	167	6153426778
1321242	Rodriguez	37134	125	231	6152431124
1321242	Rodriguez	37134	125	333	9041234445
1542311	Smithson	37134	421	125	6152439887
1542311	Smithson	37134	421	167	6153426778
1542311	Smithson	37134	421	231	6152431124
1542311	Smithson	37134	421	333	9041234445
1657399	Vanloo	32145	231	125	6152439887
1657399	Vanloo	32145	231	167	6153426778
1657399	Vanloo	32145	231	231	6152431124
1657399	Vanloo	32145	231	333	9041234445

- First, a **PRODUCT** of the tables is created

## Exercise: Step 2

How many rows  
are the  
**AGENT\_CODE**  
values are  
equal?

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1132445	Walker	32145	231	125	6152439887
1132445	Walker	32145	231	167	6153426778
1132445	Walker	32145	231	231	6152431124
1132445	Walker	32145	231	333	9041234445
1217782	Adares	32145	125	125	6152439887
1217782	Adares	32145	125	167	6153426778
1217782	Adares	32145	125	231	6152431124
1217782	Adares	32145	125	333	9041234445
1312243	Rakowski	34129	167	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1312243	Rakowski	34129	167	231	6152431124
1312243	Rakowski	34129	167	333	9041234445
1321242	Rodriguez	37134	125	125	6152439887
1321242	Rodriguez	37134	125	167	6153426778
1321242	Rodriguez	37134	125	231	6152431124
1321242	Rodriguez	37134	125	333	9041234445
1542311	Smithson	37134	421	125	6152439887
1542311	Smithson	37134	421	167	6153426778
1542311	Smithson	37134	421	231	6152431124
1542311	Smithson	37134	421	333	9041234445
1657399	Vanloo	32145	231	125	6152439887
1657399	Vanloo	32145	231	167	6153426778
1657399	Vanloo	32145	231	231	6152431124
1657399	Vanloo	32145	231	333	9041234445

## Exercise: Step 2

- Second, a SELECT is performed on the output of Step 1 to yield only the rows for which the join column(s) values are equal.

FIGURE 3.12 NATURAL JOIN, STEP 2: SELECT

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124

## Step 3

- A PROJECT is performed on the results of Step 2 to yield a single copy of each attribute, thereby eliminating duplicate columns.

FIGURE 3.13 NATURAL JOIN, STEP 3: PROJECT

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	6152439887
1321242	Rodriguez	37134	125	6152439887
1312243	Rakowski	34129	167	6153426778
1132445	Walker	32145	231	6152431124
1657399	Vanloo	32145	231	6152431124

# Types of Joins (1 of 2)

- **Natural join:** Links tables by selecting only the rows with common values in their common attributes
  - **Join columns:** Common columns
- **Equijoin:** Links tables on the basis of an equality condition that compares specified columns of each table
- **Theta join:** Extension of natural join, denoted by adding a theta subscript after the JOIN symbol





# Types of Joins (2 of 2)

- **Inner join:** Only returns matched records from the tables that are being joined (Natural, equi, theta)
- **Outer join:** Matched pairs are retained and unmatched values in the other table are left null
  - **Left outer join:** Yields all of the rows in the first table, including those that do not have a matching value in the second table
  - **Right outer join:** Yields all of the rows in the second table, including those that do not have matching values in the first table

## Figure 3.10 - Two Tables That Will Be Used in JOIN Illustrations

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

## FIGURE 3.14 LEFT OUTER JOIN

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124
1542311	Smithson	37134	421		

- A left outer join yields all of the rows in the CUSTOMER table, including those that do not have a matching value in the AGENT table.

## FIGURE 3.15 RIGHT OUTER JOIN

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124
				333	9041234445

- A right outer join yields all of the rows in the AGENT table, including those that do not have matching values in the CUSTOMER table.

# Relational Set Operators (2 of 2)

- **Divide**

- The DIVIDE operator is used to answer questions about one set of data being associated with all values of data in another set of data.
- Uses one 2-column table as the dividend and one single-column table as the divisor
- The tables must have a common column
- Output is a single column that contains all values from the second column of the dividend that are associated with every row in the divisor

# Figure 3.16 - Divide

FIGURE 3.16 DIVIDE

Dividend

P_CODE	CUS_CODE
123456	10400
123456	11501
123456	10030
123456	12550
234567	12350
234567	10040
234567	10500
234567	10030
234567	12550
345678	10400
345678	11630
345678	12550
456789	11630
567890	10600
567890	10030
567890	12550
678901	11500
678901	10400
678901	11630

DIVIDE

Divisor

P_CODE
123456
234567
567890

yields



CUS_CODE
10030
12550

A DIVIDE operation can be used to determine which customers, if any, purchased every product shown in the second table.

# Exercise - Divide

FIGURE 3.16 DIVIDE

P_CODE	CUS_CODE
123456	10400
123456	11501
123456	10030
123456	12550
234567	12350
234567	10040
234567	10500
234567	10030
234567	12550
345678	10400
345678	11630
345678	12550
456789	11630
567890	10600
567890	10030
567890	12550
678901	11500
678901	10400
678901	11630

**DIVIDE**

P_CODE
123456
234567
567890

yields

CUS_CODE
10030
12550

Why is customer 10500  
not included in the final  
result?

# Reading Task

- All the relational operators have their mathematical representations.
- Look them up in the text book.



# Data Dictionary and the System Catalog

- **Data dictionary:** Description of all tables in the database created by the user and designer
- **System catalog:** System data dictionary that describes all objects within the database
- Homonyms and synonyms must be avoided to lessen confusion
  - **Homonym:** Same name is used to label different attributes
    - C\_name both used to refer to customer name and consultant name in a database
  - **Synonym:** Different names are used to describe the same attribute
  - The data dictionary can help with avoiding homonyms and synonyms.



TABLE 3.6

## A SAMPLE DATA DICTIONARY

TABLE NAME	ATTRIBUTE NAME	CONTENTS	TYPE	FORMAT	RANGE	REQUIRED	PK OR FK	FK REFERENCED TABLE
CUSTOMER	CUS_CODE	Customer account code	CHAR(5)	99999	10000–99999	Y	PK	
	CUS_LNAME	Customer last name	VARCHAR(20)	Xxxxxxxx		Y		
	CUS_FNAME	Customer first name	VARCHAR(20)	Xxxxxxxx		Y		
	CUS_INITIAL	Customer initial	CHAR(1)	X				
	CUS_RENEW_DATE	Customer insurance renewal date	DATE	dd-mmm-yyyy				
	AGENT_CODE	Agent code	CHAR(3)	999			FK	AGENT
AGENT	AGENT_CODE	Agent code	CHAR(3)	999		Y	PK	
	AGENT_AREACODE	Agent area code	CHAR(3)	999		Y		
	AGENT_PHONE	Agent telephone number	CHAR(8)	999–9999		Y		
	AGENT_LNAME	Agent last name	VARCHAR(20)	Xxxxxxxx		Y		
	AGENT_YTD_SLS	Agent year-to-date sales	NUMBER(9,2)	9,999,999.99				

FK	= Foreign key
PK	= Primary key
CHAR	= Fixed character length data (1 – 255 characters)
VARCHAR	= Variable character length data (1 – 2,000 characters)
NUMBER	= Numeric data. NUMBER (9,2) is used to specify numbers with up to nine digits, including two digits to the right of the decimal place. Some RDBMS permit the use of a MONEY or CURRENCY data type.

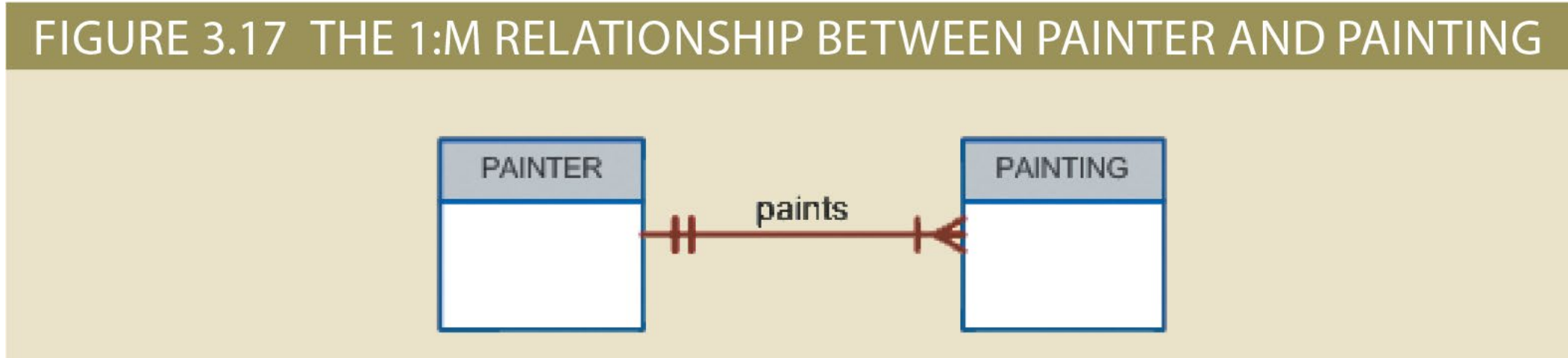
# Question: Relationships

How many types of relationships do we have in a relational database model?

# Relationships within the Relational Database

- 1:M relationship - Norm for relational databases
  - Ideal
- 1:1 relationship - One entity can be related to only one other entity and vice versa
  - Should be rare
- Many-to-many (M:N) relationship - Implemented by creating a new entity in 1:M relationships with the original entities
  - Cannot be implemented in a relational model
- **Composite entity (Bridge or associative entity):** Helps avoid problems inherent to M:N relationships, includes the primary keys of tables to be linked

## Figure 3.17 - The 1:M Relationship Between PAINTER and PAINTING



- Each painting was created by one and only one painter, but each painter could have created many paintings.
- Implemented by putting primary key of the “1” side in the table of the “many” side as a foreign key

# The 1:M Relationship Between PAINTER and PAINTING

FIGURE 3.18 THE IMPLEMENTED 1:M RELATIONSHIP BETWEEN PAINTER AND PAINTING

Table name: PAINTER

Primary key: PAINTER\_NUM

Foreign key: none

Database name: Ch03\_Museum

PAINTER_NUM	PAINTER_LNAME	PAINTER_FNAME	PAINTER_INITIAL
123	Ross	Georgette	P
126	Itero	Julio	G

Table name: PAINTING

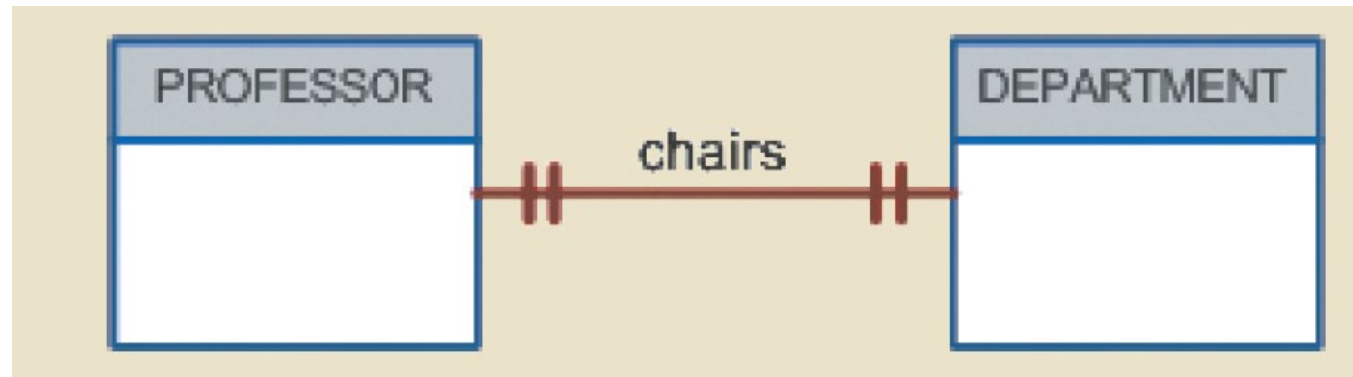
Primary key: PAINTING\_NUM

Foreign key: PAINTER\_NUM

PAINTING_NUM	PAINTING_TITLE	PAINTER_NUM
1338	Dawn Thunder	123
1339	Vanilla Roses To Nowhere	123
1340	Tired Flounders	126
1341	Hasty Exit	123
1342	Plastic Paradise	126

There is only one row in the PAINTER table for any given row in the PAINTING table, but there may be many rows in the PAINTING table for any given row in the PAINTER table.

## Figure 3.21 - The 1:1 Relationship Between PROFESSOR and DEPARTMENT





# M-N Relationships lead to Redundancies

FIGURE 3.24 THE WRONG IMPLEMENTATION OF THE M:N RELATIONSHIP BETWEEN STUDENT AND CLASS

Table name: STUDENT  
Primary key: STU\_NUM  
Foreign key: none

Database name: Ch03\_CollegeTry

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 name: CLASS  
Primary key: CLASS\_CODE  
Foreign key: STU\_NUM

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



# M-N Relationships lead to Redundancies

FIGURE 3.25 CONVERTING THE M:N RELATIONSHIP INTO TWO 1:M RELATIONSHIPS

Table name: STUDENT

Primary key: STU\_NUM

Foreign key: none

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

Database name: Ch03\_CollegeTry2

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

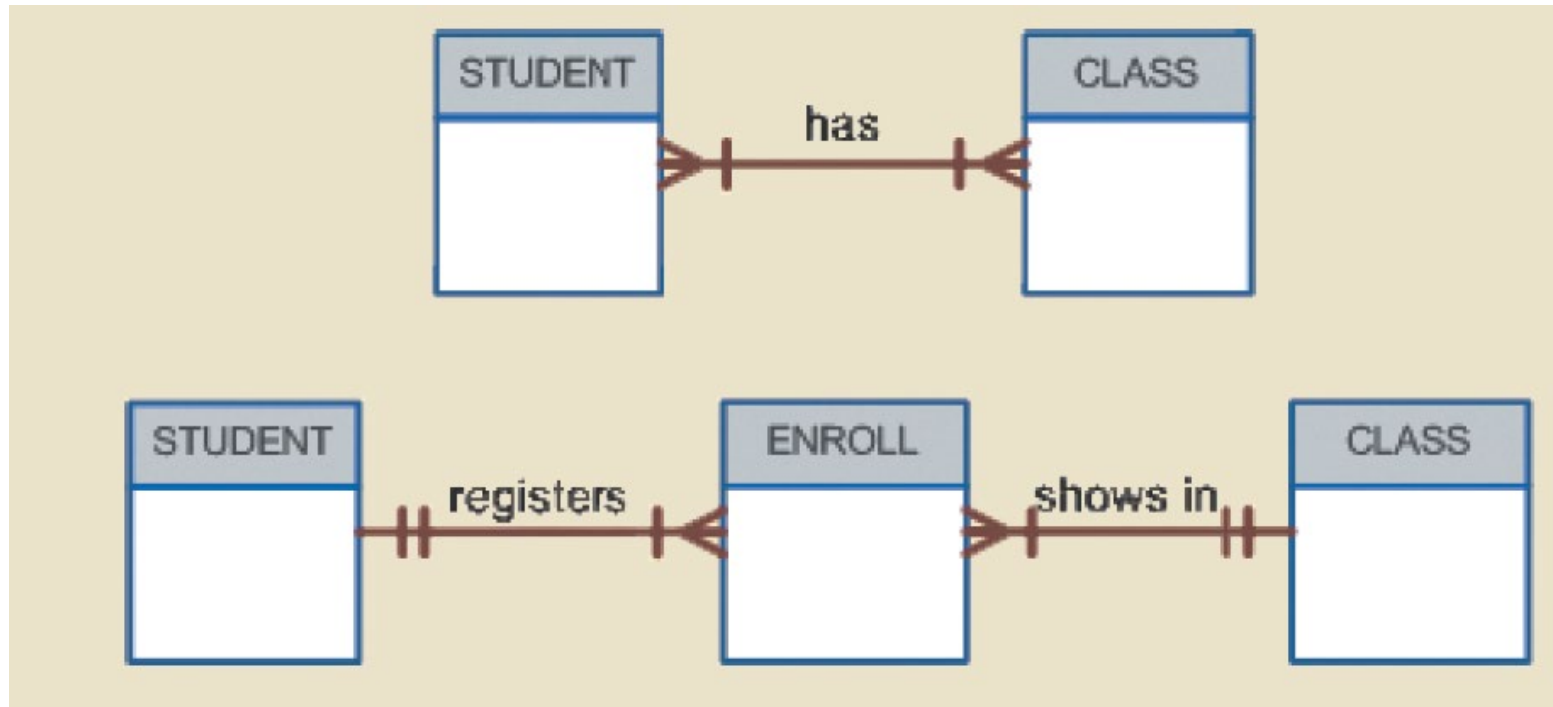
Foreign key: CRS\_CODE

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

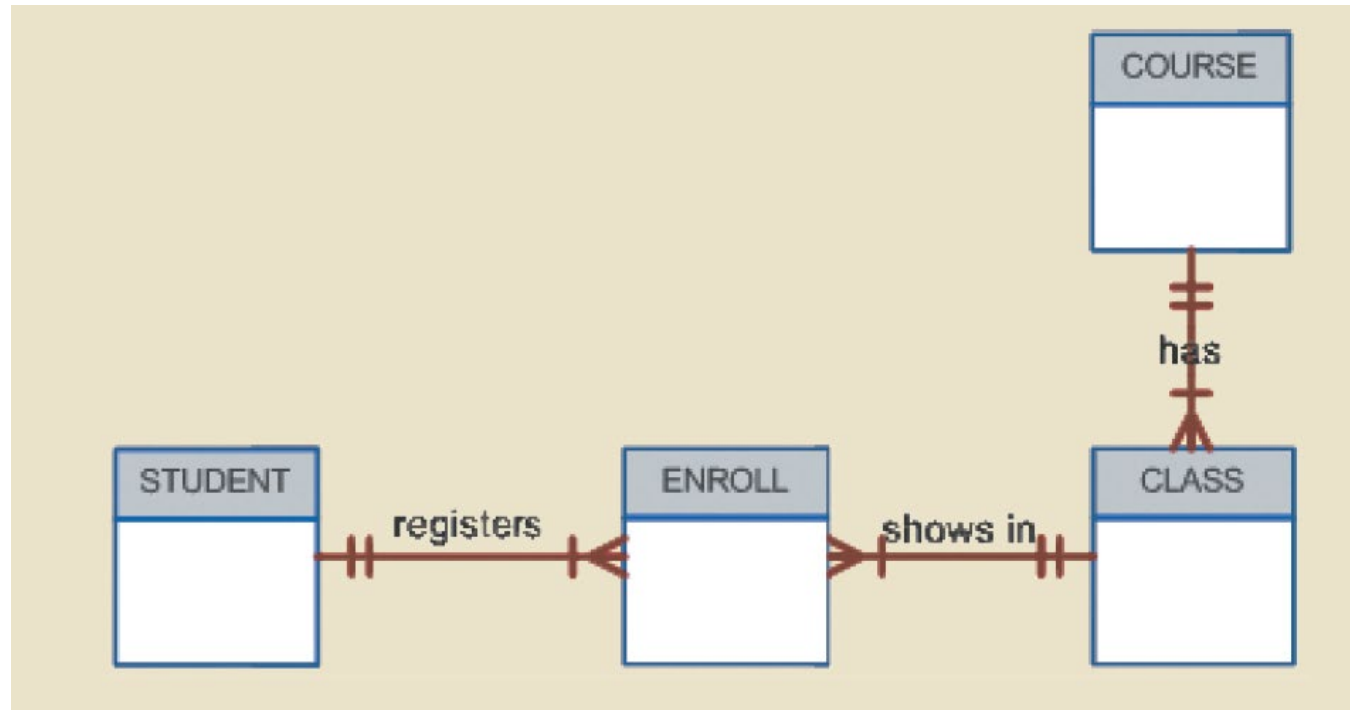
**Composite entity (Bridge or associative entity)**

**Linking Table**

## Figure 3.26 - Changing the M:N Relationship to Two 1:M Relationships



# Figure 3.27 - The Expanded ER Model



# Data Redundancy Revisited

- Relational database facilitates control of data redundancies through use of foreign keys
- To be controlled except the following circumstances
  - Data redundancy must be increased to make the database serve crucial information purposes
  - Exists to preserve the historical accuracy of the data

# Indexes

- Orderly arrangement to logically access rows in a table
- **Index key:** Index's reference point that leads to data location identified by the key
- **Unique index:** Index key can have only one pointer value associated with it
- Each index is associated with only one table