Database Management System -I

INTRODUCTION

0301203

<u>DATABASE MANAGEMENT SYSTEM - I</u>

0301206

<u>PRACTICAL ON DBMS - I</u>

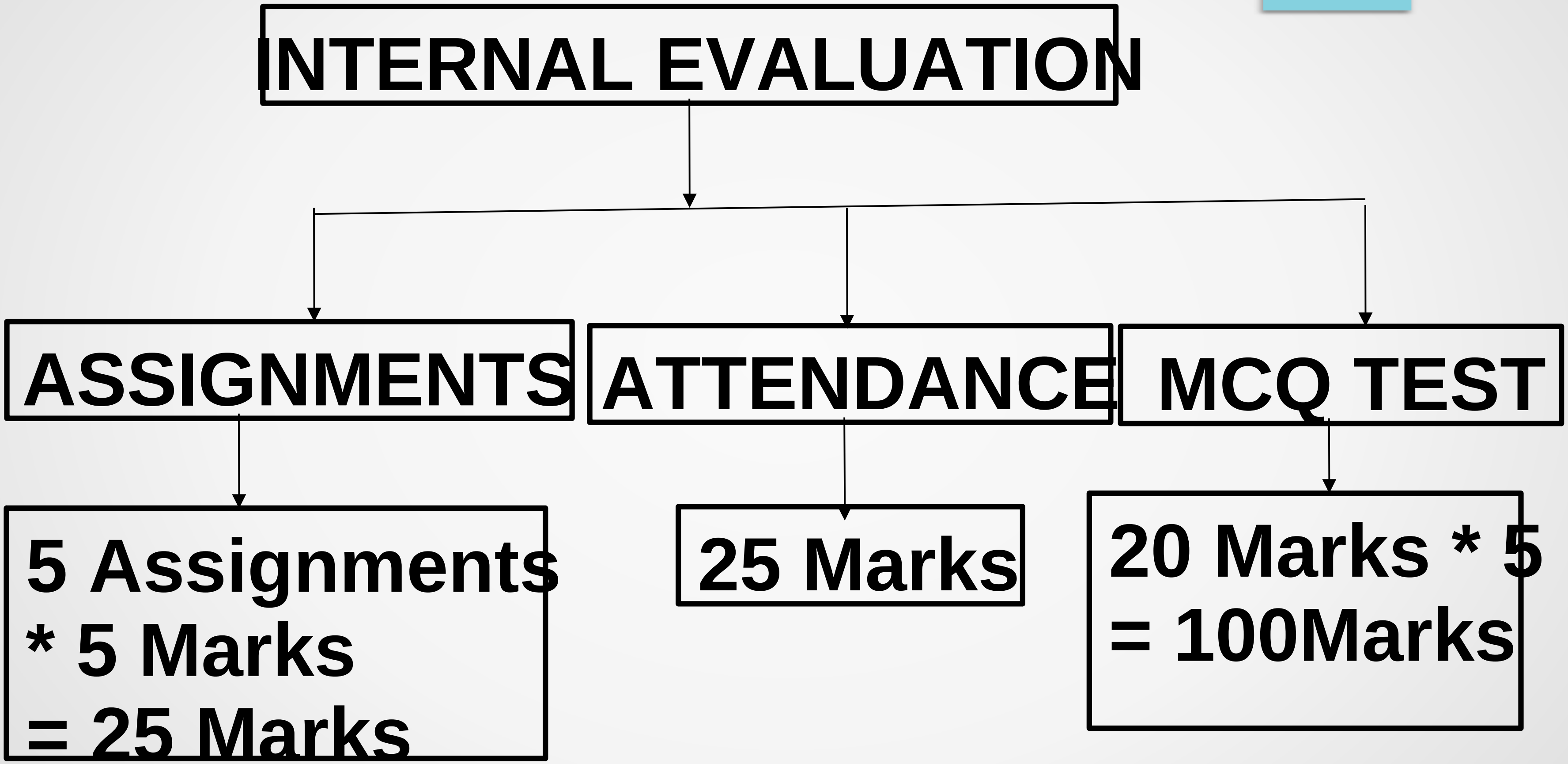
BY:

Prof. (Dr.) Ankit Bhavsar

0301203 Database Management System - I

UNIT	MODULES	WEIGHTAGE
1	Introduction to DBMS	20 %
2	Introduction to RDBMS	20 %
3	Inroduction to Normalization	20 %
4	Open Source Database Management Software	20 %

INTERNAL EVALUATION



INTERNAL EVALUATION

ASSIGNMENTS

ATTENDANCE

MCQ TEST

5 Assignments * 5 Marks = 25 Marks

25 Marks

20 Marks * 5 = 100Marks

UNIT - 2 Introduction to RDBMS

- Introduction of RDBMS
- Characteristics of RDBMS
- Componenets of RDBMS
- Relational Set Operators
- Introduction to Enitity Relationship Model

Introduction of RDMS

RDBMS stands for Relational Database Management System.

- **Most popular** database system.
- **Simple and sound** theoretical basis.
- The **model is based on tables, rows and columns** and the manipulation of data stored within.
- **Relational database is a collection of these tables.**
- First commercial system: **MULTICS in 1978.**
- **Has overtaken Hierarchical and Network models.**
- **Main feature:** Single database can be spread across several tables.
- Examples include: **Oracle, IBM's DB2, Sybase, MySQL & Microsoft Access.**

Characteristics Of RDBMS

- RDBMS is **design to handle frequently changing data.**
- It often **used in transaction oriented application.**
- RDBMS **can store vast amount of historical data**,which can later analysed.
- The RDBMS is **configured as per application** basis and a u**nique schema exist to support each application.**
- RDBMS features **ensures data intigrity.**

Components of RDBMS

- Keys
- Table and their characteristics
- Integrity Rules
- Typs of Relationship

Keys

- In Relation Model, **keys are important** because they are **used to ensure that each row in a table must be uniquely identifiable.**
- Key are **also used to establish relationships among tables** and to ensure the integrity of the data.
- **A Key** consist of one or more **attributes** that determine other attributes.
  - i.e an invoice number identifies all of the invoice attributes such as invoice date and the customer name.

Keys

- The role of the key is based on the concept of **determination.**
- The Determination is the **state in which knowing the value of one attribute makes it possible to determine the value of another.**
- I.e : ***revenue – cost = profit***
  - This is a form of determination, because if you are given the *revenue* and the *cost* , you can be determine the *profit.*

Types of Keys

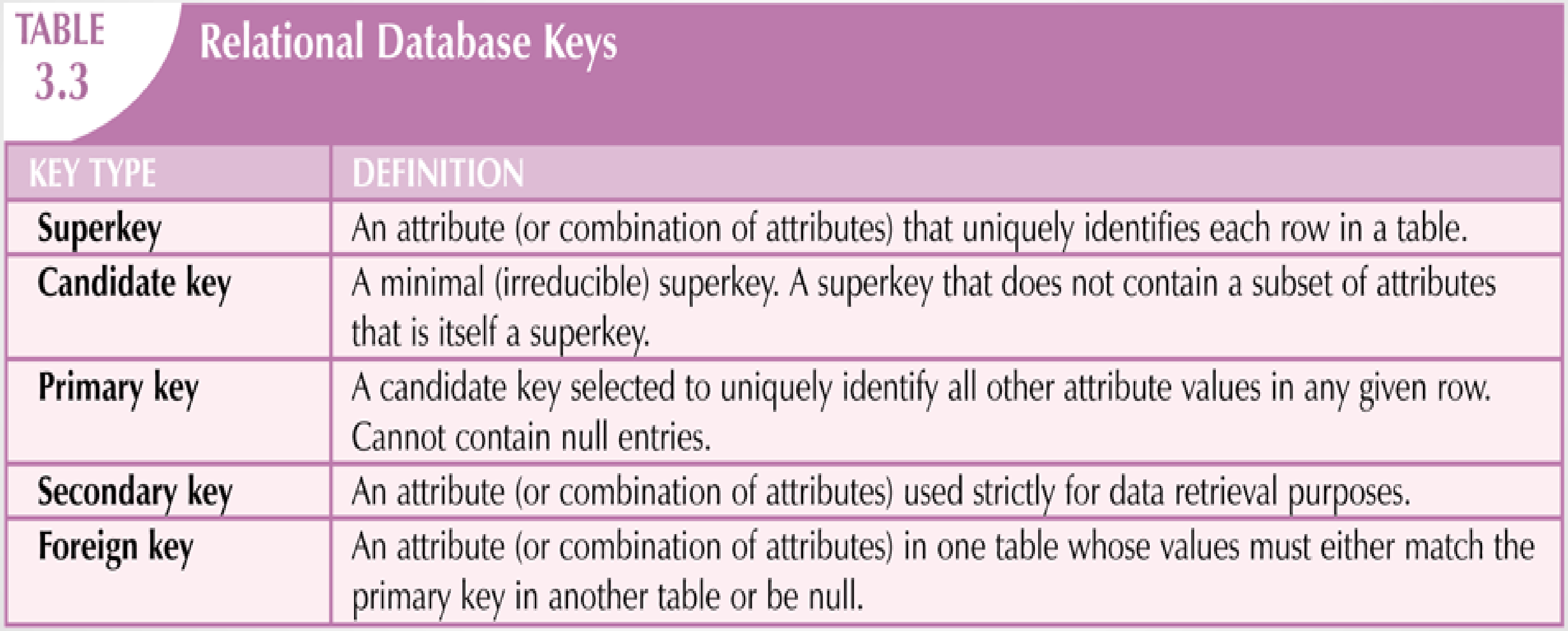


TABLE 3.3 Relational Database Keys

KEY TYPE	DEFINITION
Superkey	An attribute (or combination of attributes) that uniquely identifies each row in a table.
Candidate key	A minimal (irreducible) superkey. A superkey that does not contain a subset of attributes that is itself a superkey.
Primary key	A candidate key selected to uniquely identify all other attribute values in any given row. Cannot contain null entries.
Secondary key	An attribute (or combination of attributes) used strictly for data retrieval purposes.
Foreign key	An attribute (or combination of attributes) in one table whose values must either match the primary key in another table or be null.

Tables and Their Characteristics

- **Logical view of relational database** is facilitated by the creation of data relationships based on a logical construct known as a **relation.**
- Relation is a mathematical construct, end users find it much easier to **think of a relation as a table.**

Tables and Their Characteristics

- **Table:** two-dimensional structure composed of rows and columns ,Persistent representation of logical relation**.**

- **Table is also called a relation,** because the relation model's creator, used the two terms as synonyms.

- As far as the table's user is concerned, a table contains a group of related entity occurrence is also called entity set.
- i.e STUDENT table contains a collection of entity occurrences, each representing a student

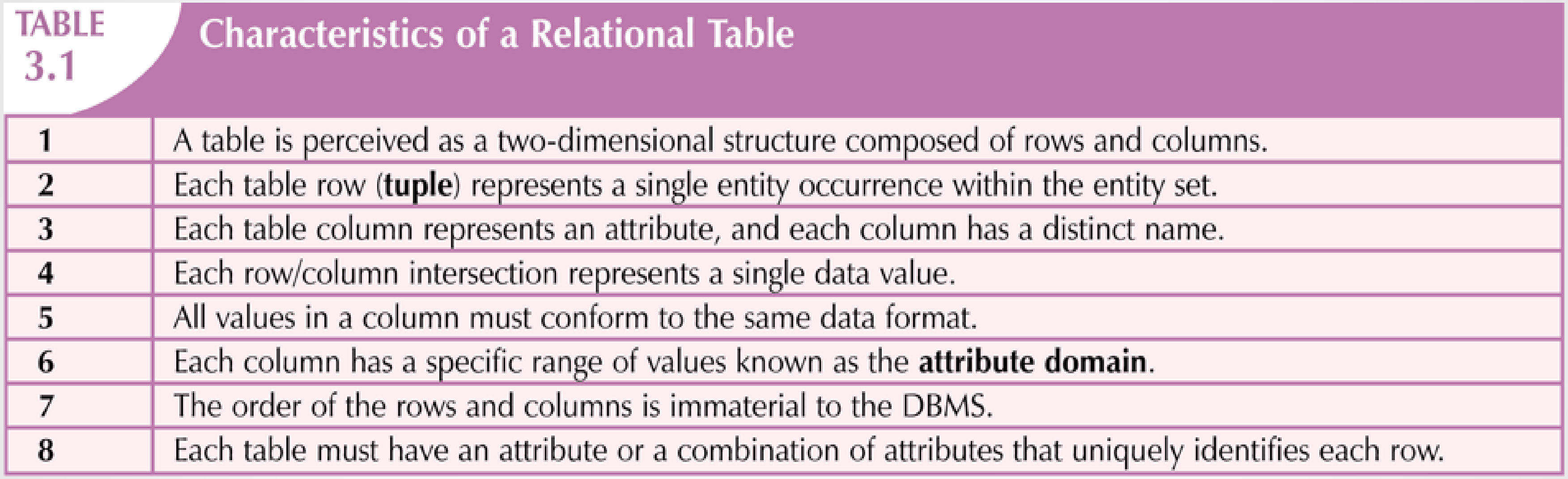


TABLE 3.1 Characteristics of a Relational Table

1	A table is perceived as a two-dimensional structure composed of rows and columns.
2	Each table row (**tuple**) represents a single entity occurrence within the entity set.
3	Each table column represents an attribute, and each column has a distinct name.
4	Each row/column intersection represents a single data value.
5	All values in a column must conform to the same data format.
6	Each column has a specific range of values known as the **attribute domain**.
7	The order of the rows and columns is immaterial to the DBMS.
8	Each table must have an attribute or a combination of attributes that uniquely identifies each row.

Types of Relationship

Relationships within the Relational Database

- 1:M relationship

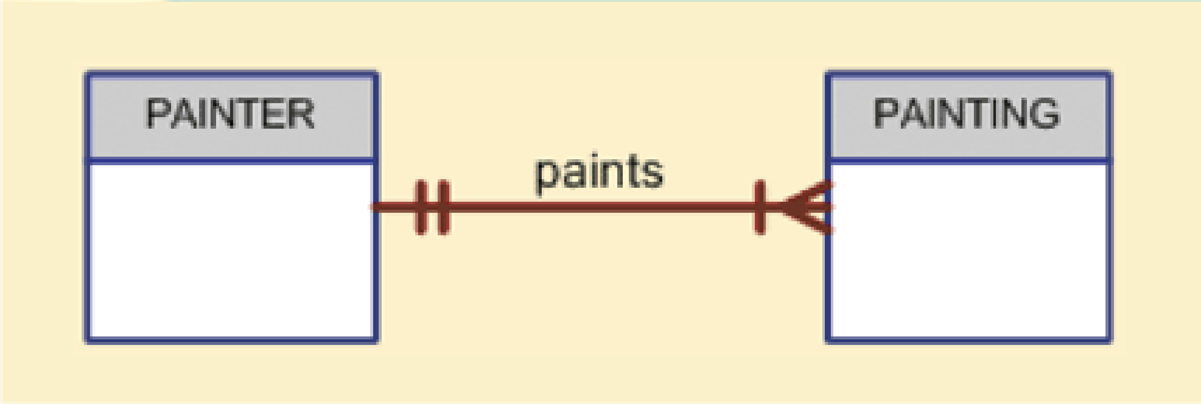
- 1:1 relationship

M:M relationship

The 1:M Relationship

- Relational database norm
- Found in any database environment

FIGURE 3.18 The 1:M relationship between PAINTER and PAINTING



The 1:M Relationship

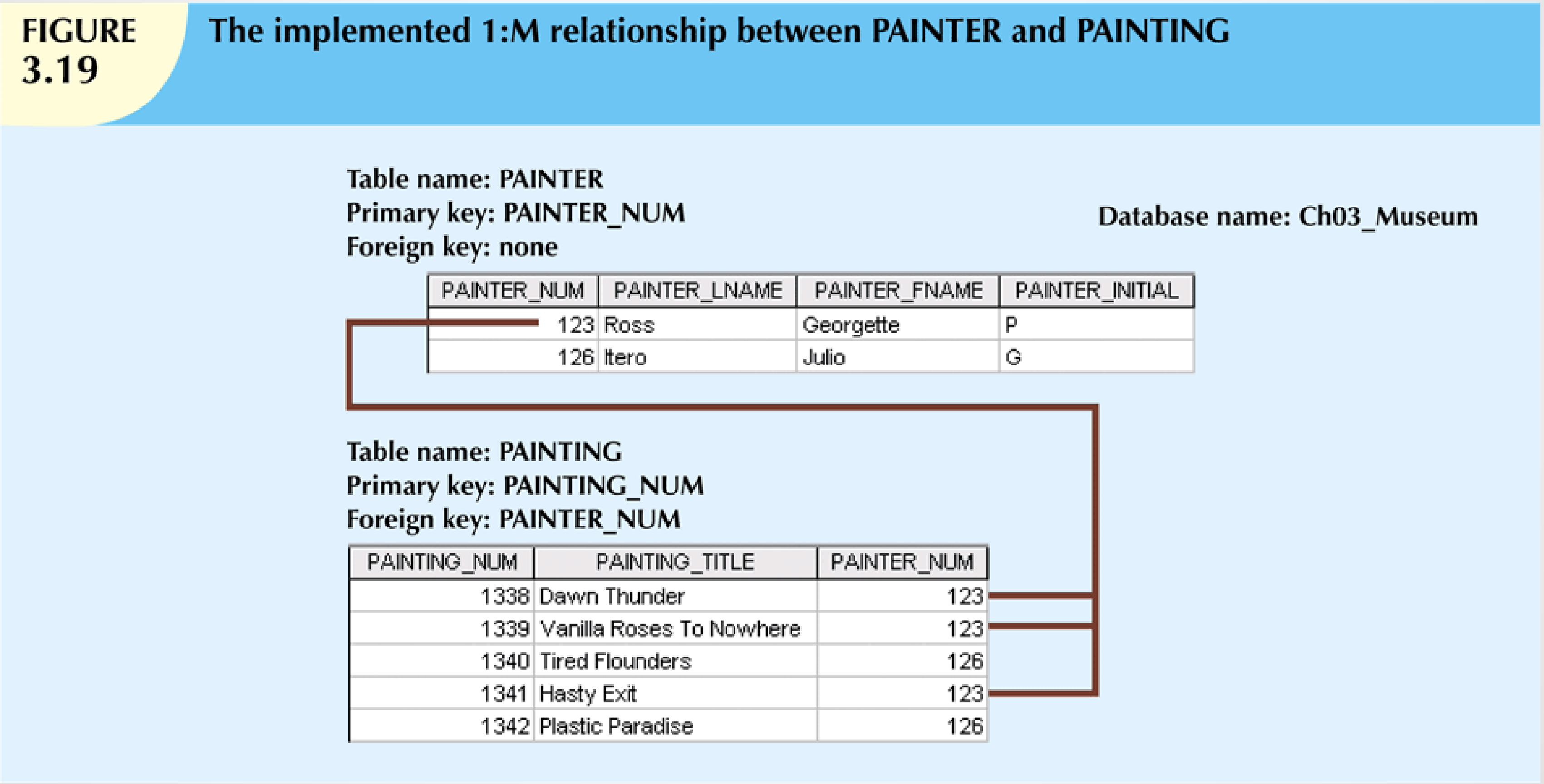


FIGURE 3.19 The implemented 1:M relationship between PAINTER and PAINTING

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

The 1:M Relationship

Eg:

- Each painting is painted by **one** and only one painter, but each painter could have painted **many** paintings.

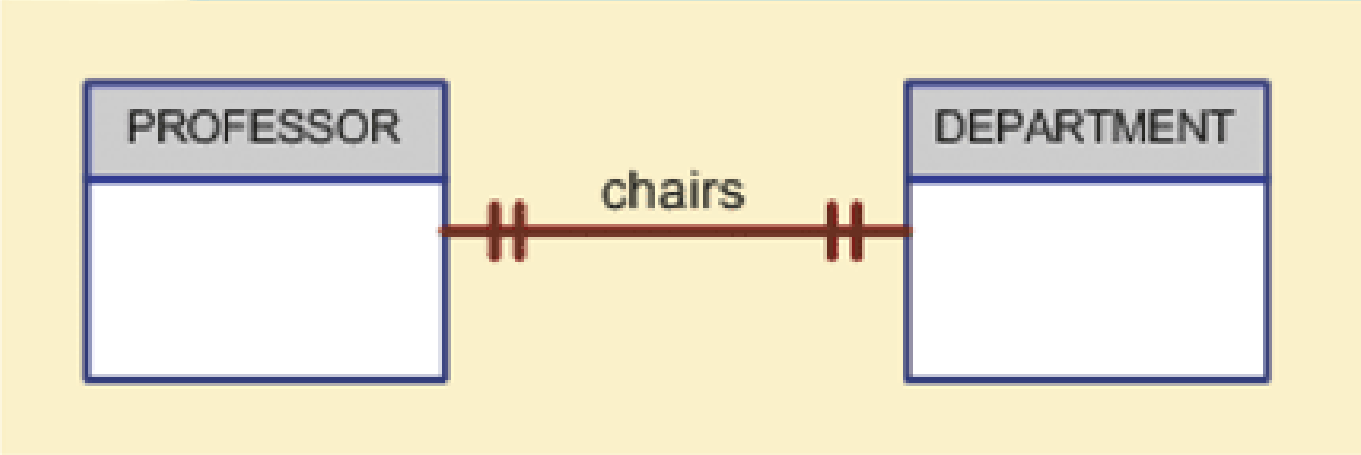
- A mother can have **many** children and each child has only **one** mother.

- A car and its parts. Each part belongs to **one** car and one car has **multiple** parts.

- Houses in a street. One street had multiple houses and a house belongs to one street.

The 1:1 Relationship

FIGURE 3.22 The 1:1 relationship between PROFESSOR and DEPARTMENT



The 1:1 Relationship

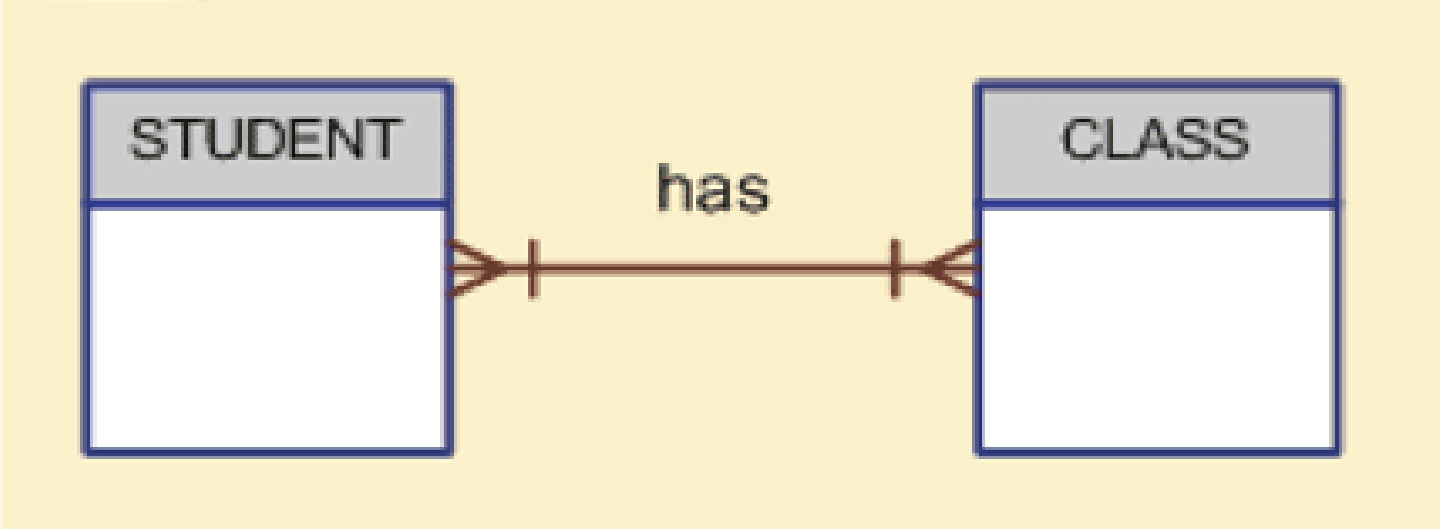
- One entity related to only one other entity, and vice versa

Eg:

- The entities **PROFESSOR CHAIR** and **DEPARTMENT** thus exhibit a 1:1 relationship.
- **One** person has **one** passport.
- **One** car model is made by **one** company.
- **One** pair of jeans has **one** brand name.
- **One** employee belongs to **one** organization.

The M:M Relationship

FIGURE 3.24 **The ERM's M:N relationship between STUDENT and CLASS**



The M:M Relationship

- Implemented by breaking it up to produce a set of 1:M relationships
- Avoid problems inherent to M:M relationship by creating a **composite entity**
  - Includes as foreign keys the primary keys of tables to be linked

The M:M Relationship

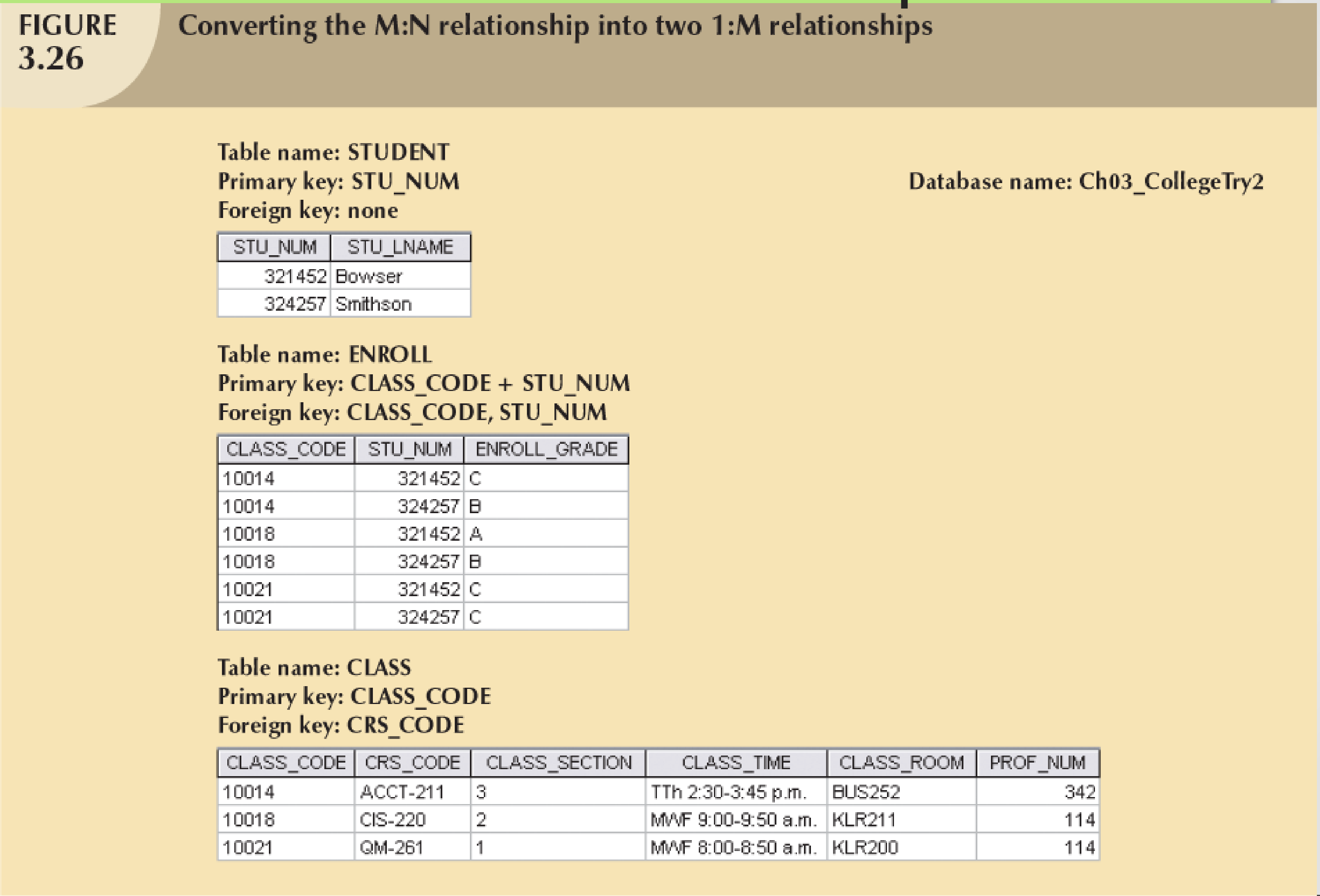


FIGURE 3.26 Converting the M:N relationship into two 1:M relationships

Database name: Ch03_CollegeTry2

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

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

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

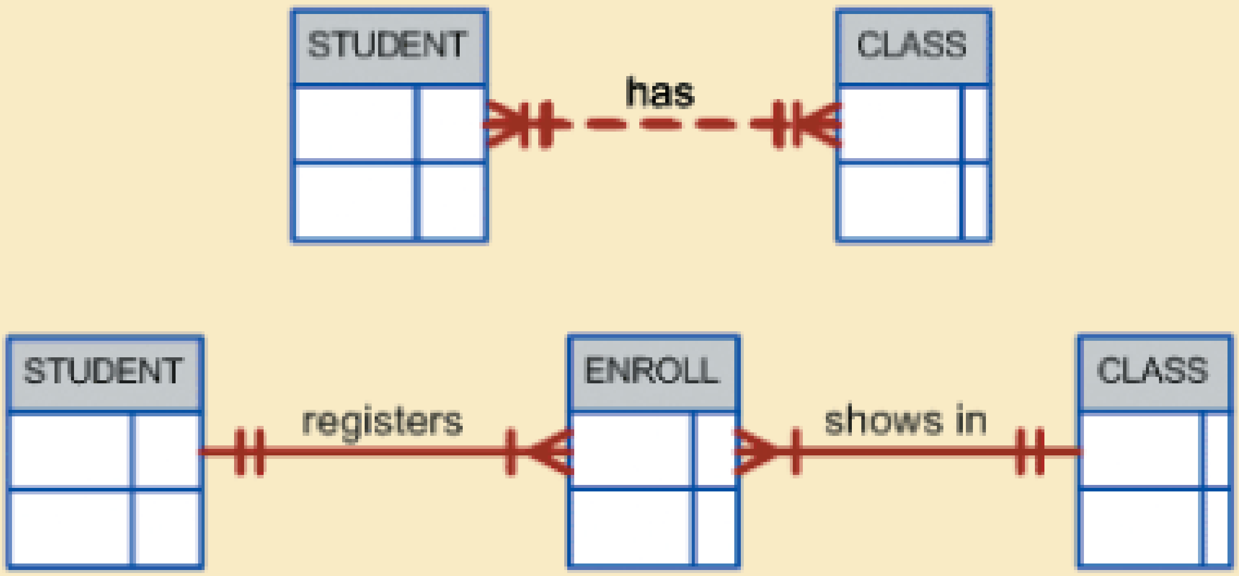
CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

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

The M:M Relationship

FIGURE 3.27 Changing the M:N relationship to two 1:M relationships



Integrity Rules

- Relational database integrity rules are very important to good database design.
- Integrity Rules are:
  - *Entity Integrity*
  - *Referential Integrity*

Integrity Rules

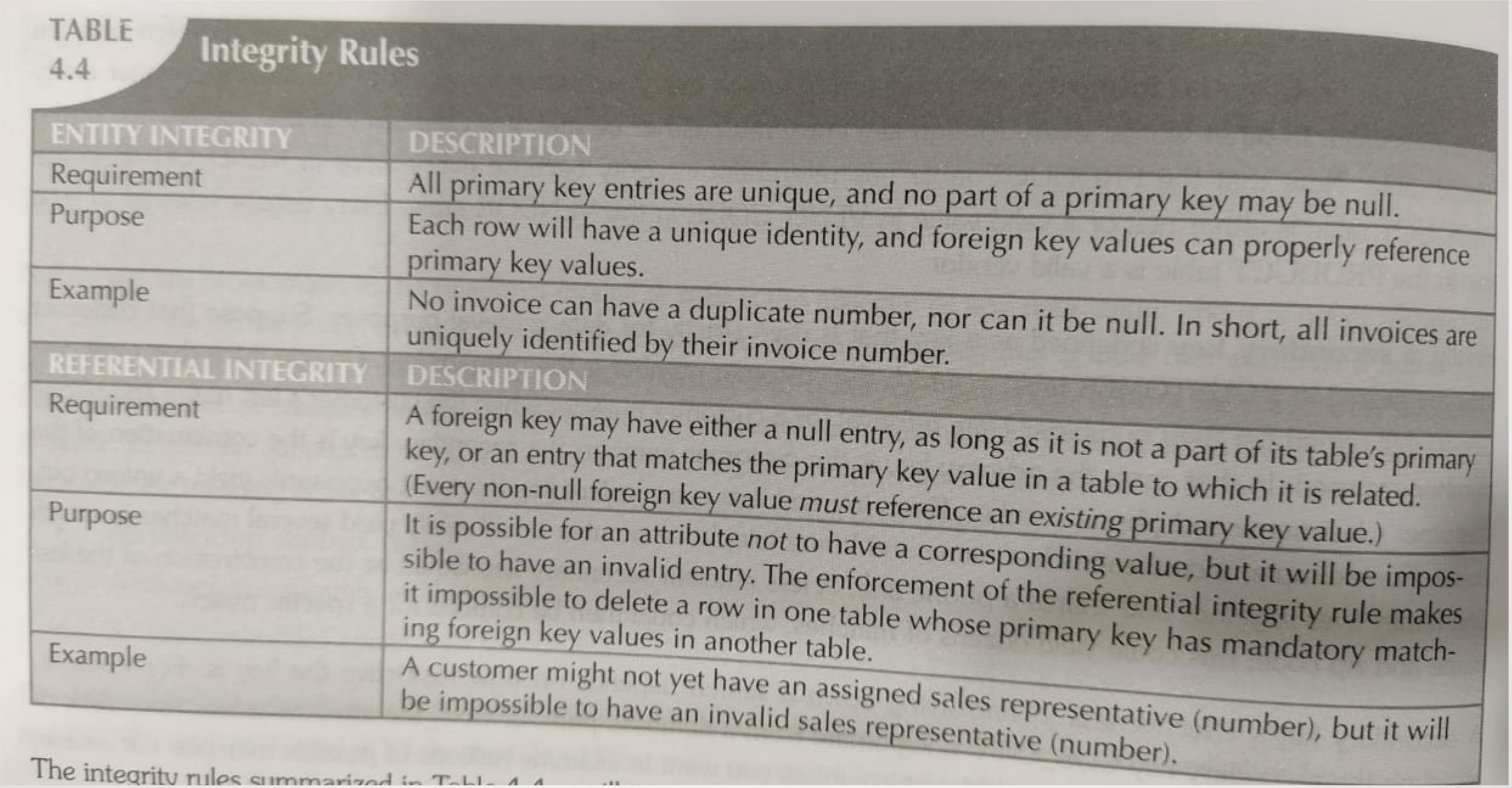
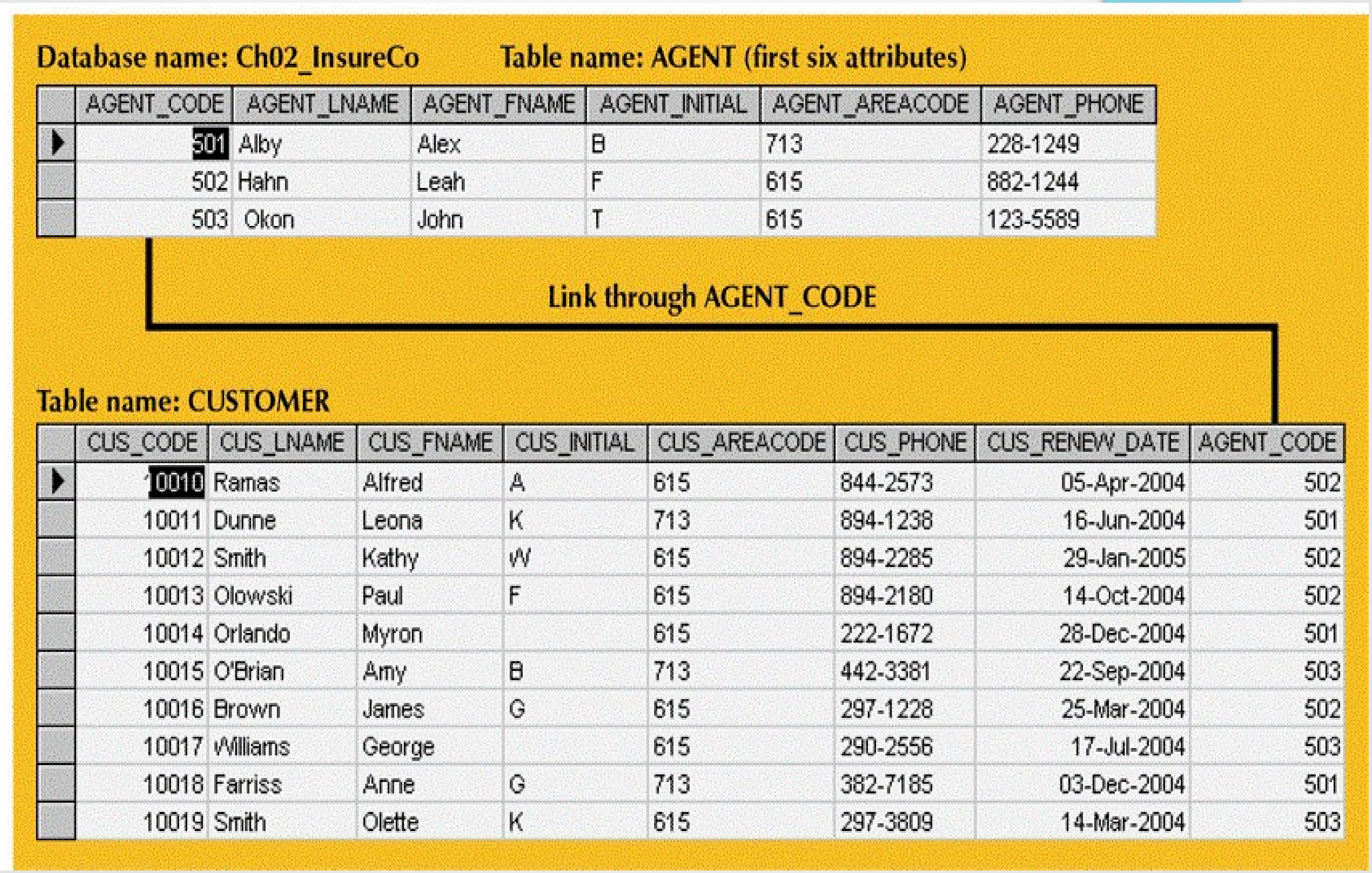


TABLE 4.4 Integrity Rules

ENTITY INTEGRITY	DESCRIPTION
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	No invoice can have a duplicate number, nor can it be null. In short, all invoices are uniquely identified by their invoice number.
REFERENTIAL INTEGRITY	**DESCRIPTION**
Requirement	A foreign key may have either a null entry, as long as it is not a part of its table's primary key, or an entry that matches the primary key value in a table to which it is related. (Every non-null foreign key value *must* reference an *existing* primary key value.)
Purpose	It is possible for an attribute *not* to have a corresponding value, but it will be impossible to have an invalid entry. The enforcement of the referential integrity rule makes it impossible to delete a row in one table whose primary key has mandatory matching foreign key values in another table.
Example	A customer might not yet have an assigned sales representative (number), but it will be impossible to have an invalid sales representative (number).

The integrity rules summarized in Table 4.4 will

Integrity Rules



Database name: Ch02_InsureCo

Table name: AGENT (first six attributes)

AGENT_CODE	AGENT_LNAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE
501	Alby	Alex	B	713	228-1249
502	Hahn	Leah	F	615	882-1244
503	Okon	John	T	615	123-5589

Link through AGENT_CODE

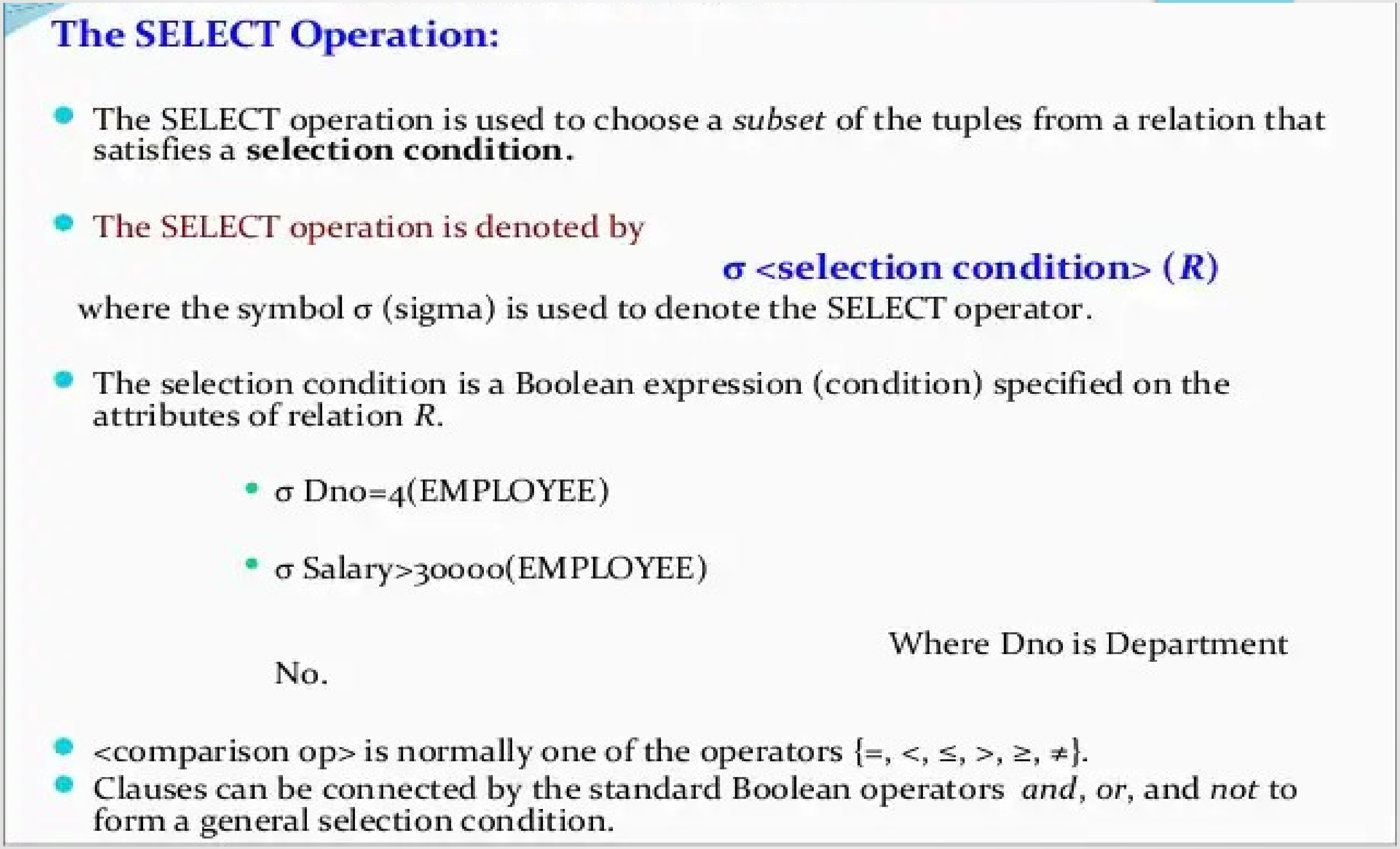
Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	615	844-2573	05-Apr-2004	502
10011	Dunne	Leona	K	713	894-1238	16-Jun-2004	501
10012	Smith	Kathy	W	615	894-2285	29-Jan-2005	502
10013	Olowski	Paul	F	615	894-2180	14-Oct-2004	502
10014	Orlando	Myron		615	222-1672	28-Dec-2004	501
10015	O'Brian	Amy	B	713	442-3381	22-Sep-2004	503
10016	Brown	James	G	615	297-1228	25-Mar-2004	502
10017	Williams	George		615	290-2556	17-Jul-2004	503
10018	Farriss	Anne	G	713	382-7185	03-Dec-2004	501
10019	Smith	Olette	K	615	297-3809	14-Mar-2004	503

Relational Set Operators

- DBMS supports relational set operators.
- The major relational set operators are :
  - Select
  - Project
  - Union
  - Intersection
  - Difference
  - Cartesian Product
- All of these can be implemented in DBMS using different queries.

Relational Set Operator - Select



The SELECT Operation:

- The SELECT operation is used to choose a *subset* of the tuples from a relation that satisfies a **selection condition.**

- The SELECT operation is denoted by

  σ <selection condition> (*R*)

  where the symbol σ (sigma) is used to denote the SELECT operator.

- The selection condition is a Boolean expression (condition) specified on the attributes of relation *R*.

  - σ Dno=4(EMPLOYEE)

  - σ Salary>30000(EMPLOYEE)

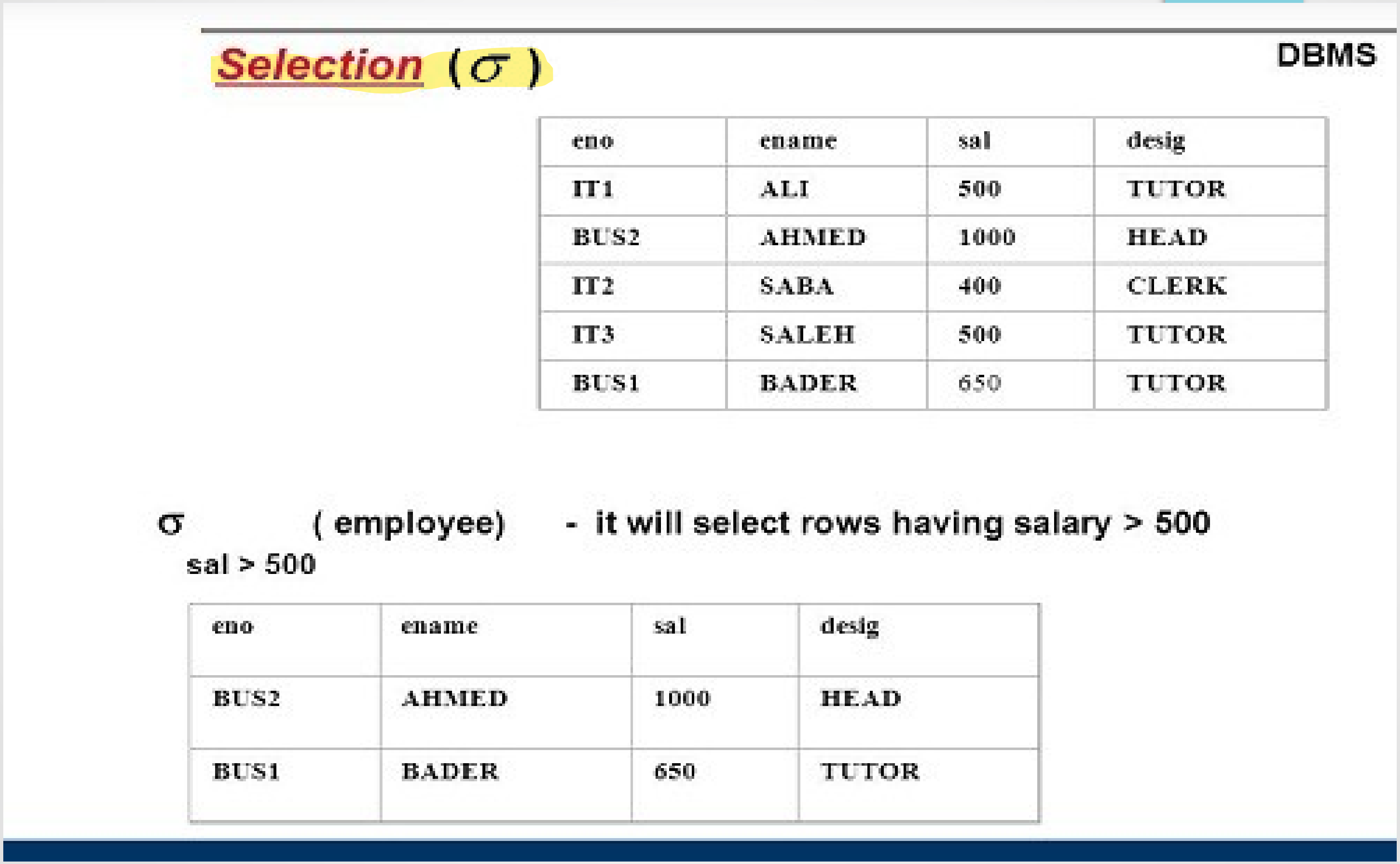
  Where Dno is Department No.

- <comparison op> is normally one of the operators {=, <, ≤, >, ≥, ≠}.
- Clauses can be connected by the standard Boolean operators *and*, *or*, and *not* to form a general selection condition.

Relational Set Operator - Select

- Select operation **chooses the subset of tuples from the relation that satisfies the given condition mentioned in the syntax** of selection. The selection operation is also **known as horizontal partitioning** since it partitions the table or relation horizontally.
- Notation: *σ c(R)*
  - where 'c' is selection condition which is a boolean expression(condition)
- ie.
  - *SELECT * FROM employee where salary > 500;*

Relational Set Operator - Select



Selection (σ)

DBMS

eno	ename	sal	desig
IT1	ALI	500	TUTOR
BUS2	AHMED	1000	HEAD
IT2	SABA	400	CLERK
IT3	SALEH	500	TUTOR
BUS1	BADER	650	TUTOR

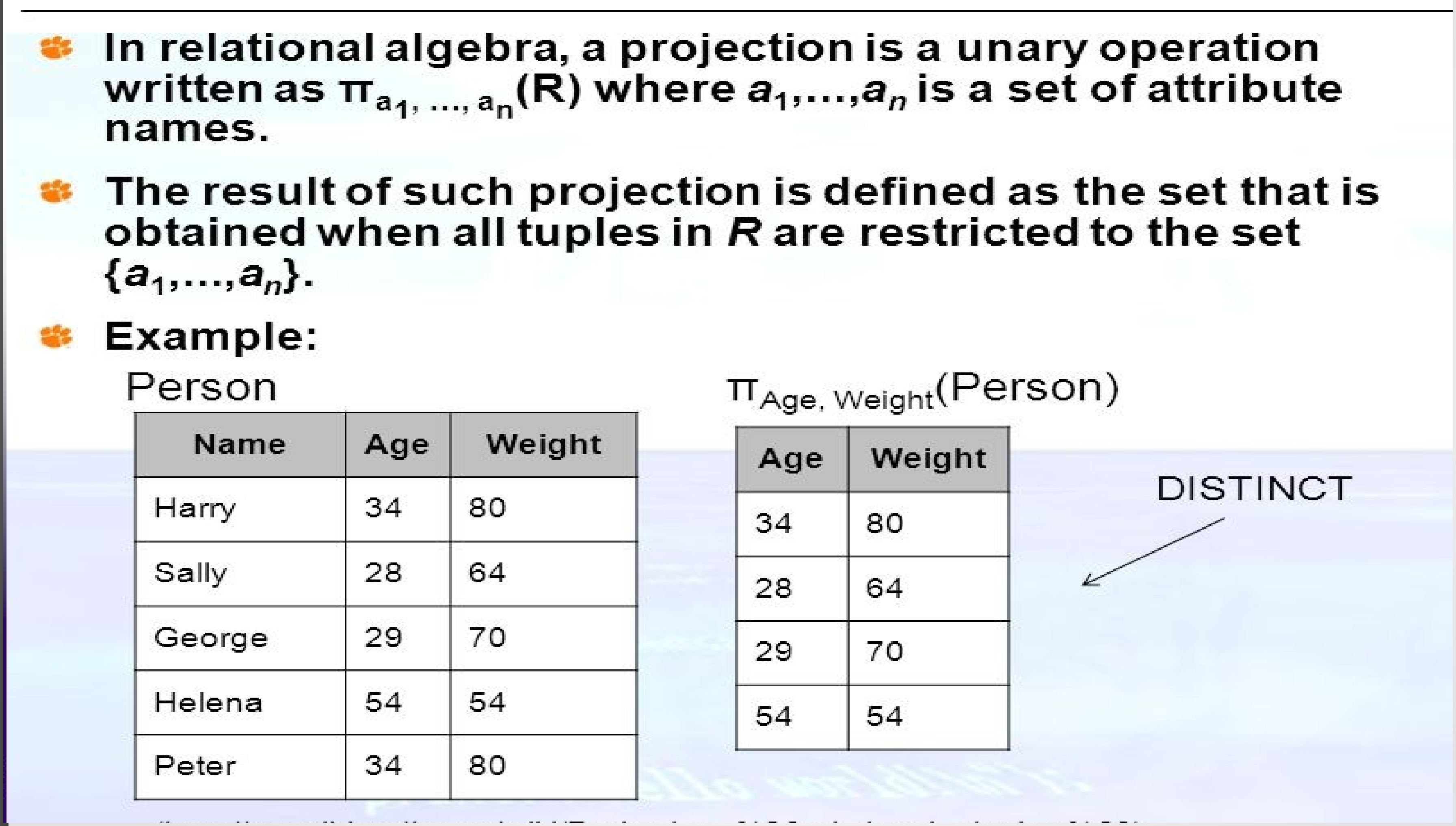
$\sigma_{sal > 500}$ (employee) - it will select rows having salary > 500

eno	ename	sal	desig
BUS2	AHMED	1000	HEAD
BUS1	BADER	650	TUTOR

Relational Set Operator - Project

- It displays the **specific column of a table**.
- It is denoted by *pie (∏).*
- It is a vertical subset of the original relation. **It eliminates duplicate tuples.**
- The syntax is as follows − ∏*regno(student)*

Relational Set Operator - Project



- In relational algebra, a projection is a unary operation written as $\pi_{a_1, \ldots, a_n}(R)$ where $a_1,\ldots,a_n$ is a set of attribute names.
- The result of such projection is defined as the set that is obtained when all tuples in *R* are restricted to the set $\{a_1,\ldots,a_n\}$.
- Example:

Person

Name	Age	Weight
Harry	34	80
Sally	28	64
George	29	70
Helena	54	54
Peter	34	80

$\pi_{\text{Age, Weight}}(\text{Person})$

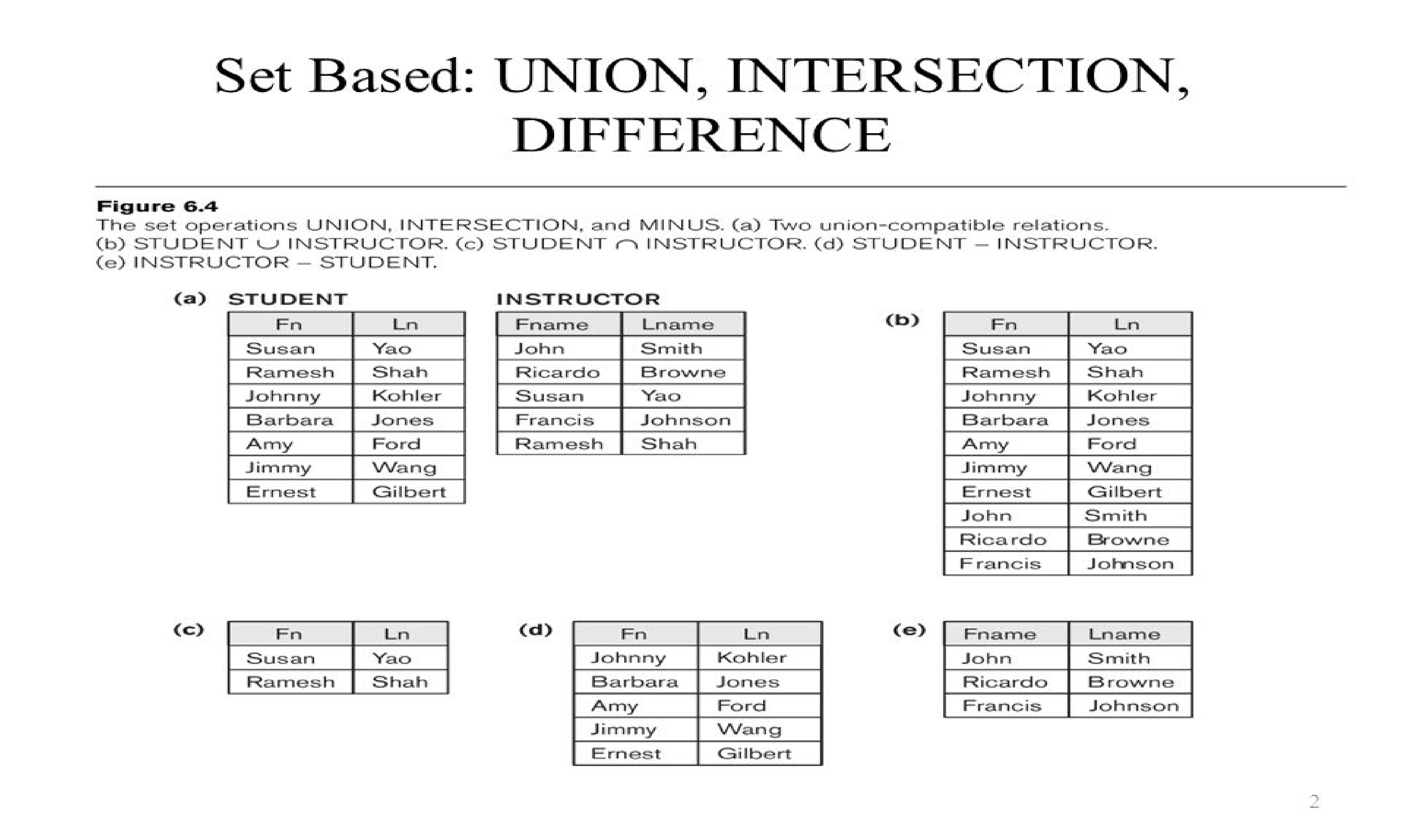
Age	Weight
34	80
28	64
29	70
54	54

DISTINCT

Relational Set Operator

- **Union**
  - Union **combines two different results** obtained by a query into a single result in the form of a table.
- **Intersection**
  - The intersection operator gives the **common data values between the two data** sets that are intersected.
- **Difference**
  - The set difference operators takes the two sets and **returns the values that are in the first set but not the second set.**

Relational Set Operator



Set Based: UNION, INTERSECTION, DIFFERENCE

Figure 6.4
The set operations UNION, INTERSECTION, and MINUS. (a) Two union-compatible relations. (b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT − INSTRUCTOR. (e) INSTRUCTOR − STUDENT.

(a) **STUDENT**

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

(b)

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

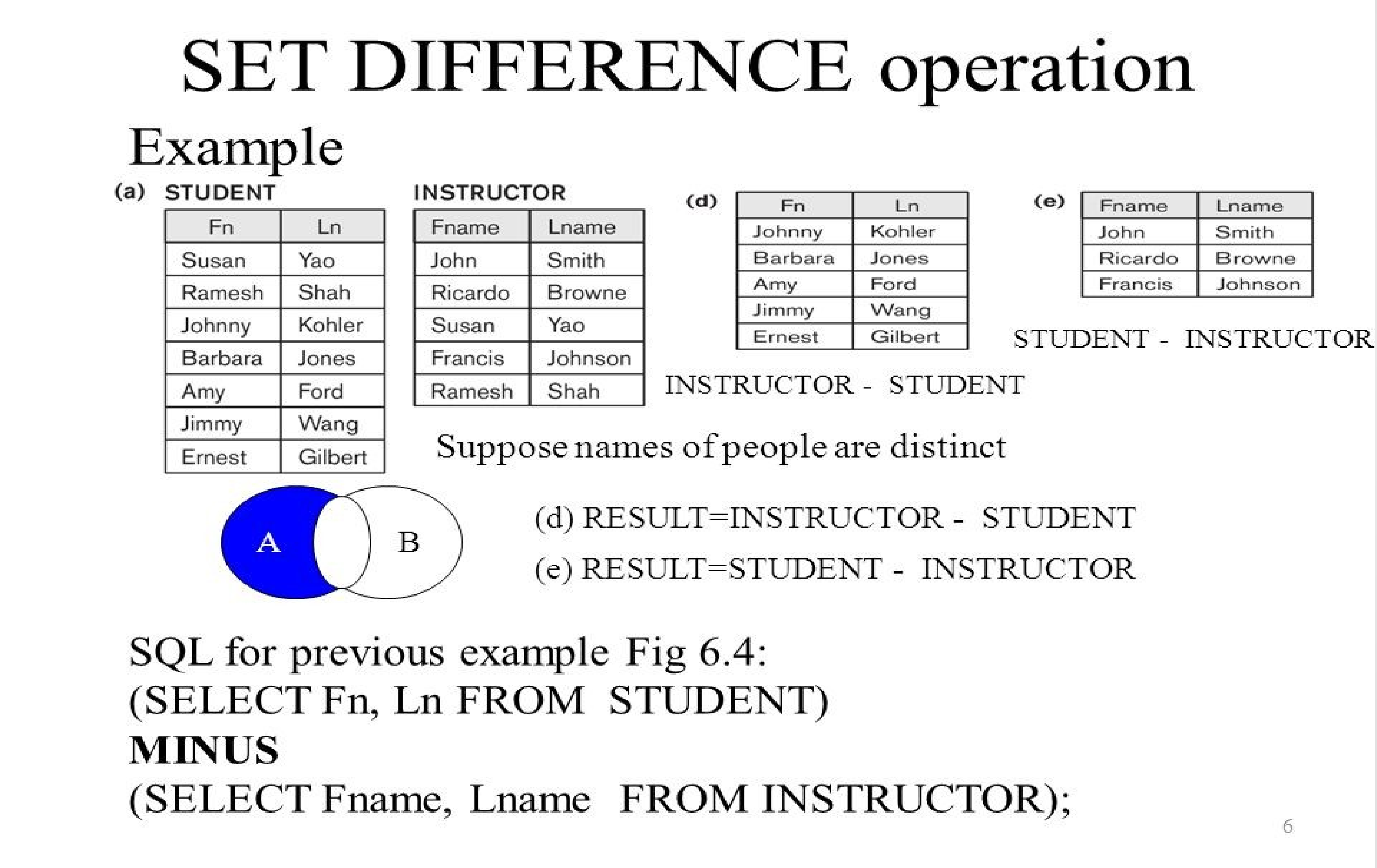
(d)

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

Relational Set Operator



SET DIFFERENCE operation

Example

(a) STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

(d)

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR - STUDENT

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

STUDENT - INSTRUCTOR

Suppose names of people are distinct

(d) RESULT=INSTRUCTOR - STUDENT

(e) RESULT=STUDENT - INSTRUCTOR

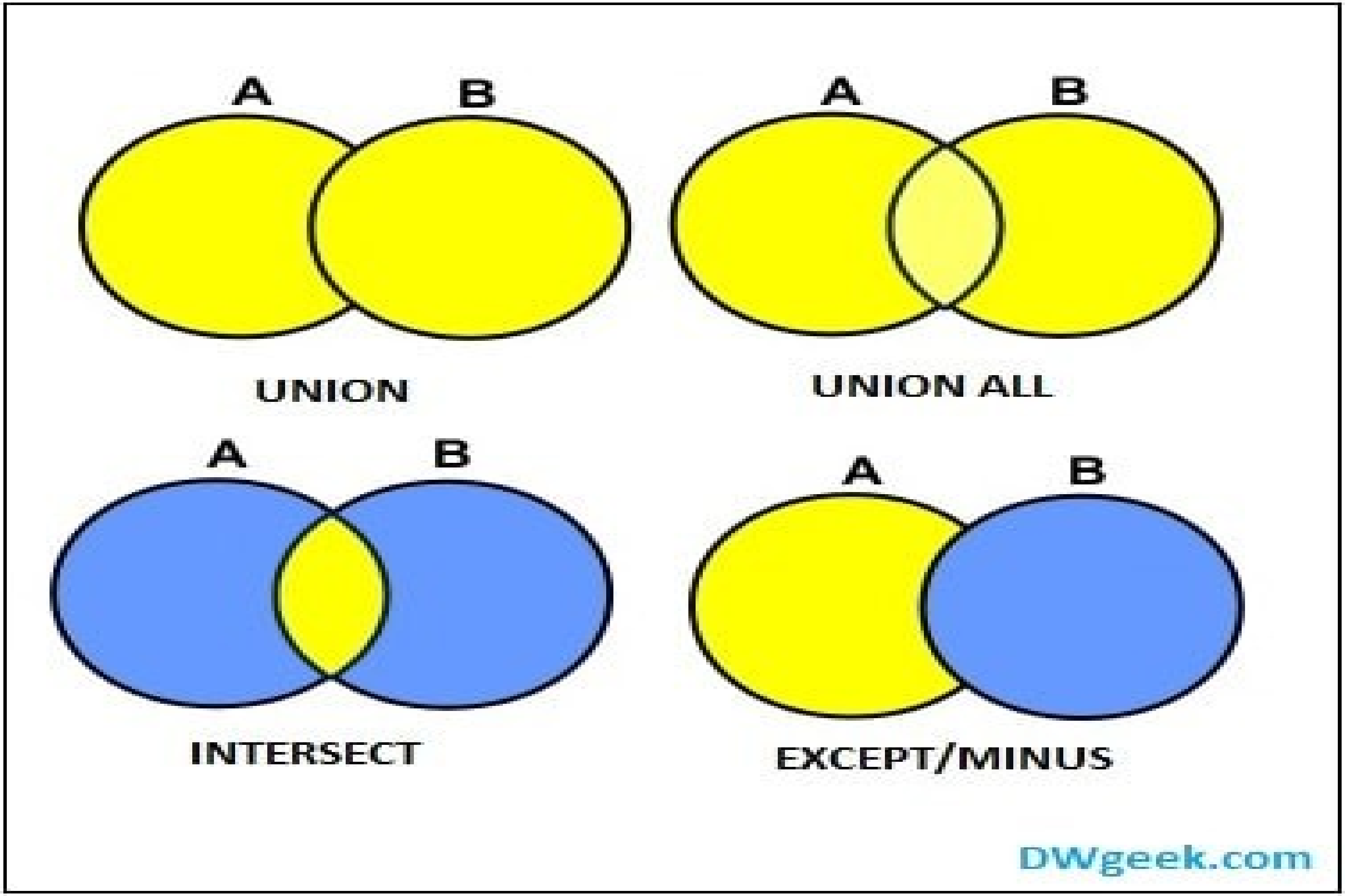
SQL for previous example Fig 6.4:

(SELECT Fn, Ln FROM STUDENT)

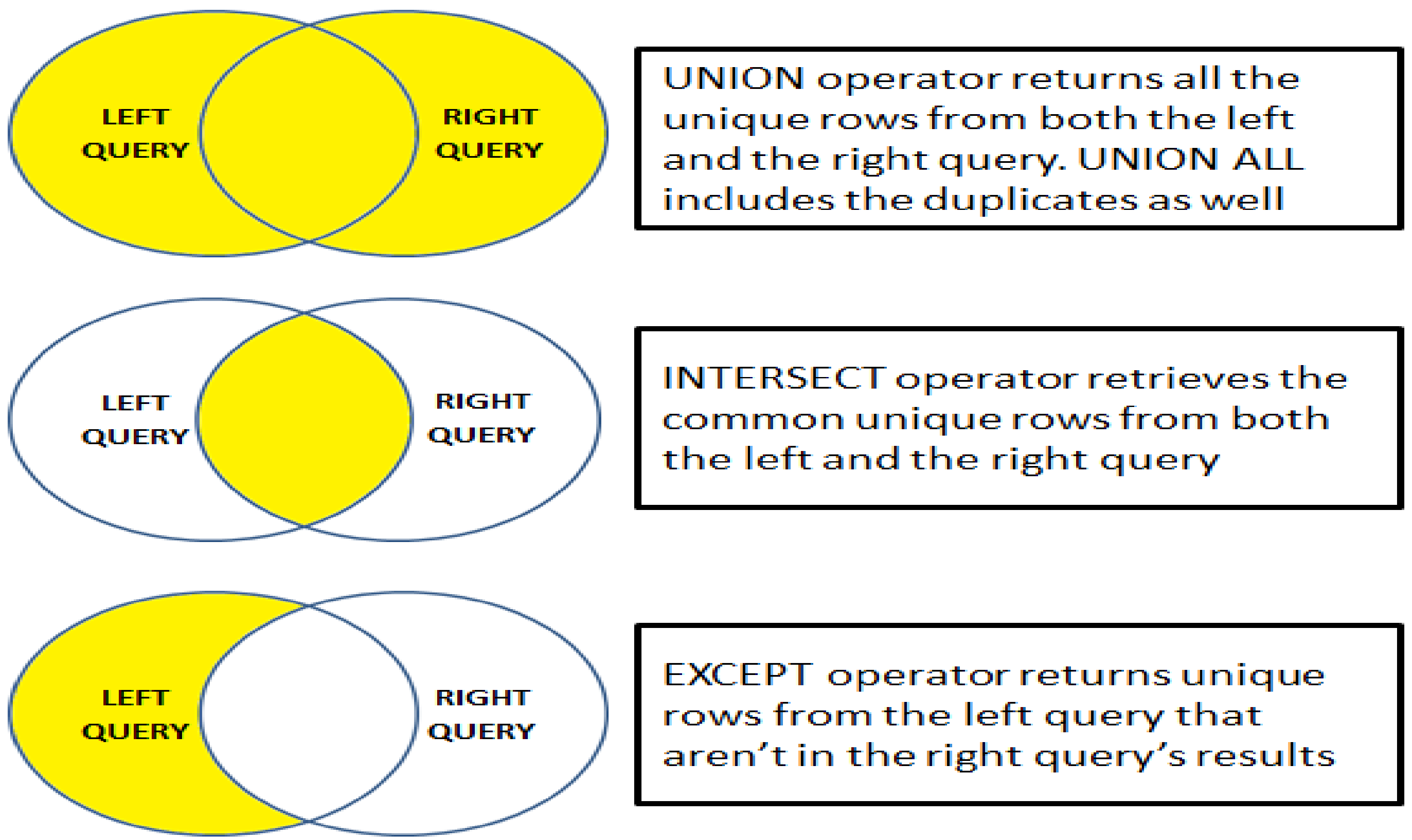
MINUS

(SELECT Fname, Lname FROM INSTRUCTOR);

Relational Set Operator



Relational Set Operator



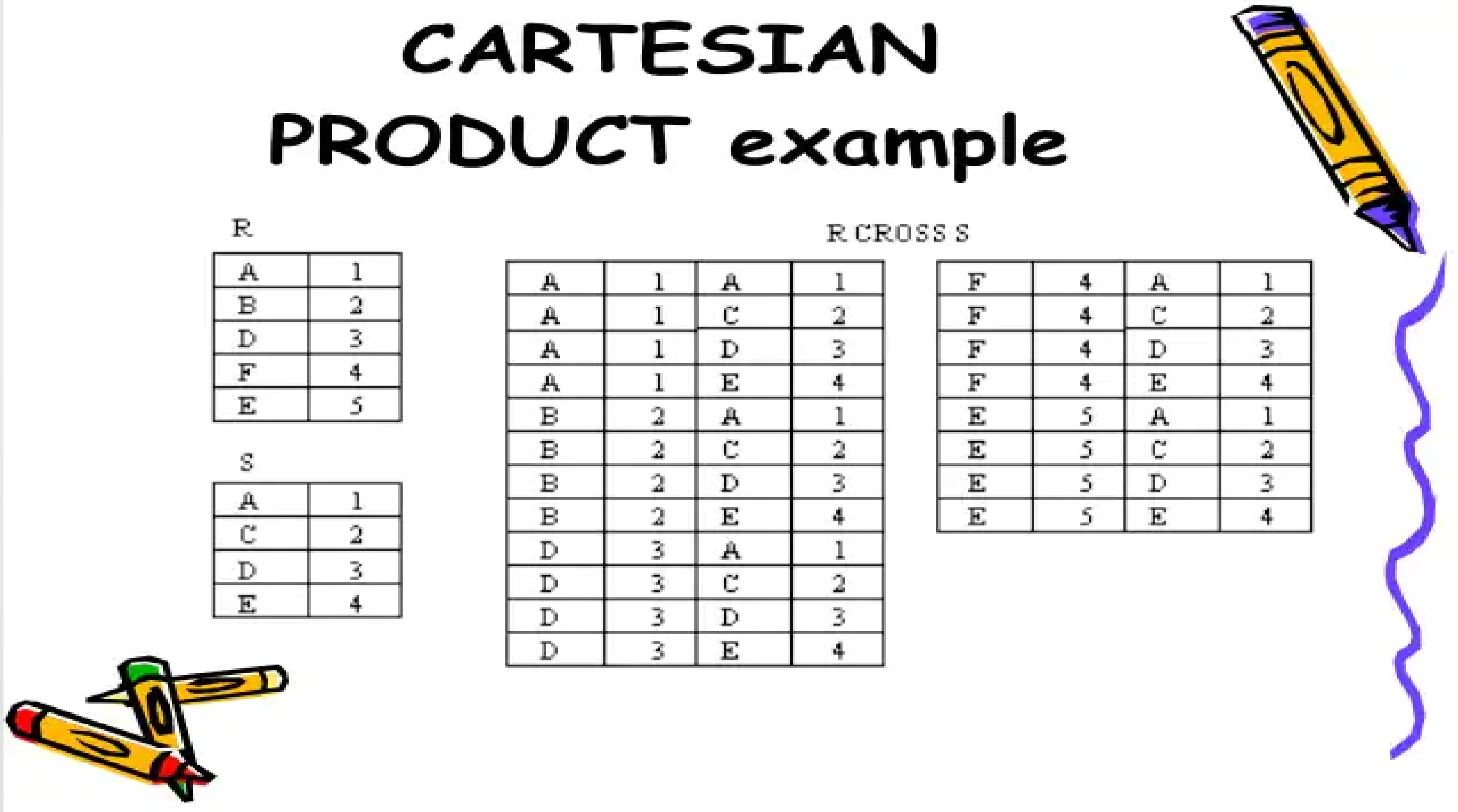
UNION operator returns all the unique rows from both the left and the right query. UNION ALL includes the duplicates as well

INTERSECT operator retrieves the common unique rows from both the left and the right query

EXCEPT operator returns unique rows from the left query that aren’t in the right query’s results

Relational Set Operator

- **Cartesian Product**



CARTESIAN PRODUCT example

R

A	1
B	2
D	3
F	4
E	5

S

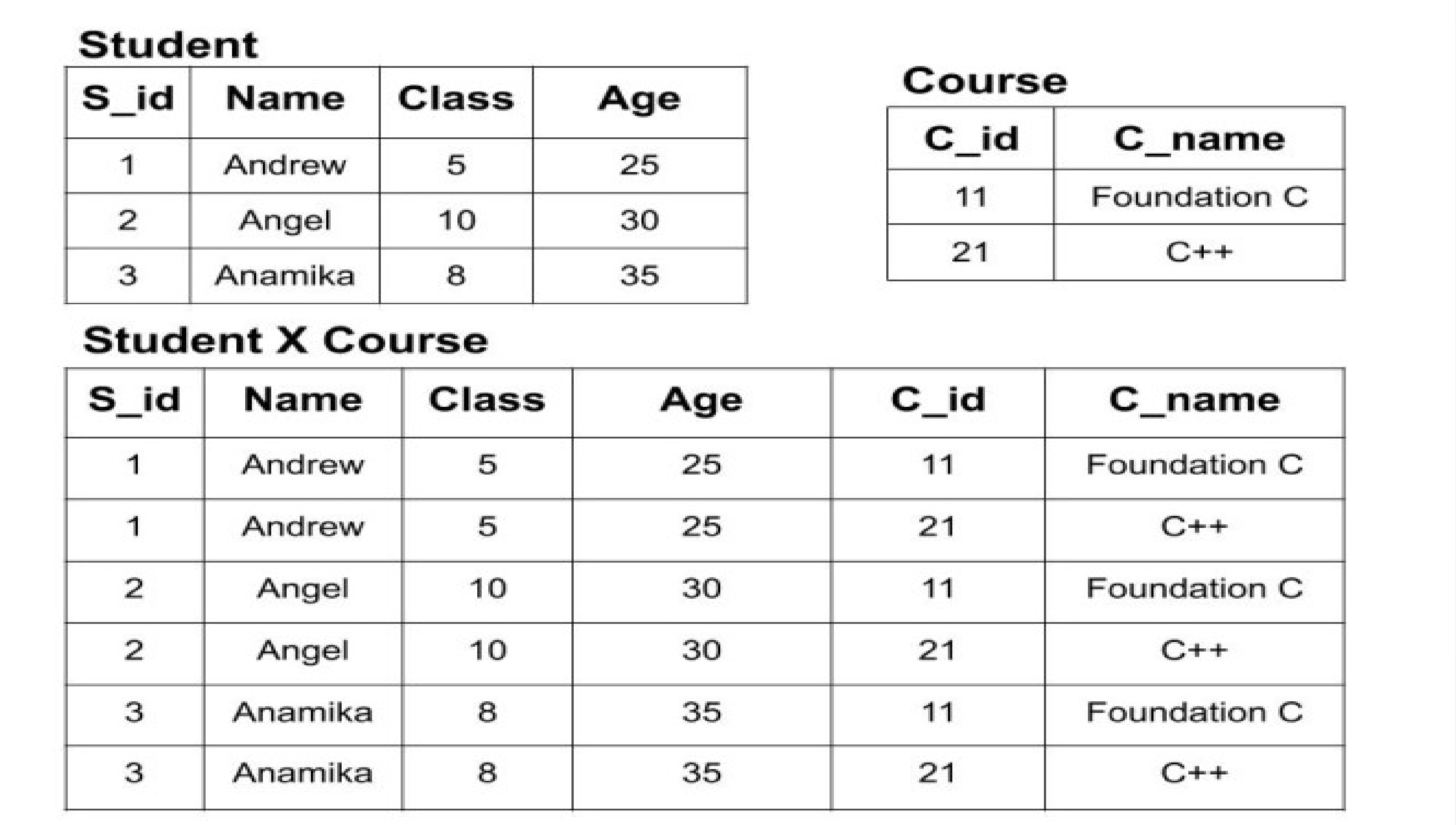
A	1
C	2
D	3
E	4

R CROSS S

A	1	A	1
A	1	C	2
A	1	D	3
A	1	E	4
B	2	A	1
B	2	C	2
B	2	D	3
B	2	E	4
D	3	A	1
D	3	C	2
D	3	D	3
D	3	E	4
F	4	A	1
F	4	C	2
F	4	D	3
F	4	E	4
E	5	A	1
E	5	C	2
E	5	D	3
E	5	E	4

Relational Set Operator

- **Cartesian Product**



Student

S_id	Name	Class	Age
1	Andrew	5	25
2	Angel	10	30
3	Anamika	8	35

Course

C_id	C_name
11	Foundation C
21	C++

Student X Course

S_id	Name	Class	Age	C_id	C_name
1	Andrew	5	25	11	Foundation C
1	Andrew	5	25	21	C++
2	Angel	10	30	11	Foundation C
2	Angel	10	30	21	C++
3	Anamika	8	35	11	Foundation C
3	Anamika	8	35	21	C++