Database Systems

(CS 355 / CE 373)

Dr. Umer Tariq
Assistant Professor,
Dhanani School of Science & Engineering,
Habib University

Acknowledgements

- Many slides have been borrowed from the official lecture slides accompanying the textbook:

  Database System Concepts, (2019), Seventh Edition,
  Avi Silberschatz, Henry F. Korth, S. Sudarshan
  McGraw-Hill, ISBN 9780078022159

  The original lecture slides are available at:

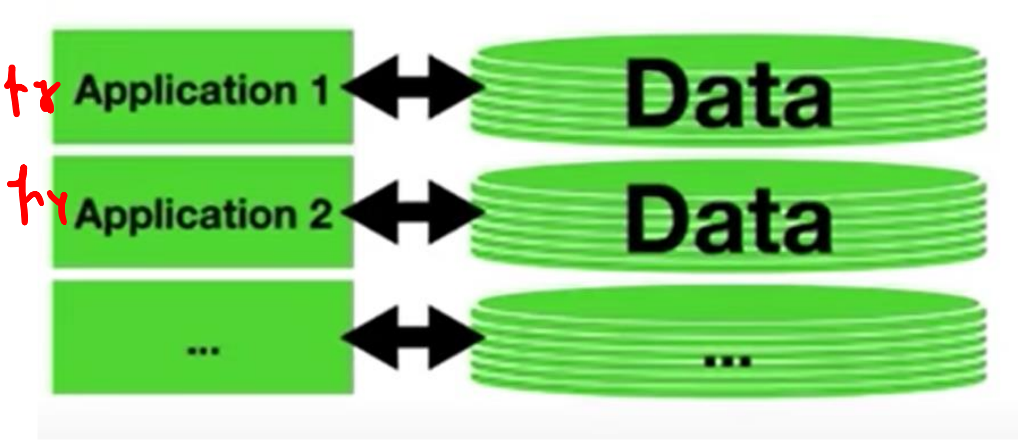
  https://www.db-book.com/

- Some of the slides have been borrowed from the lectures by Dr. Immanuel Trummer (Cornell University). Available at: (www.itrummer.org)

Outline: Week 2

- Introduction to Relational Data Model
- Structure of Relational Databases
- Relational Database Schema
- Keys
- Schema Diagrams

File-Based Approach



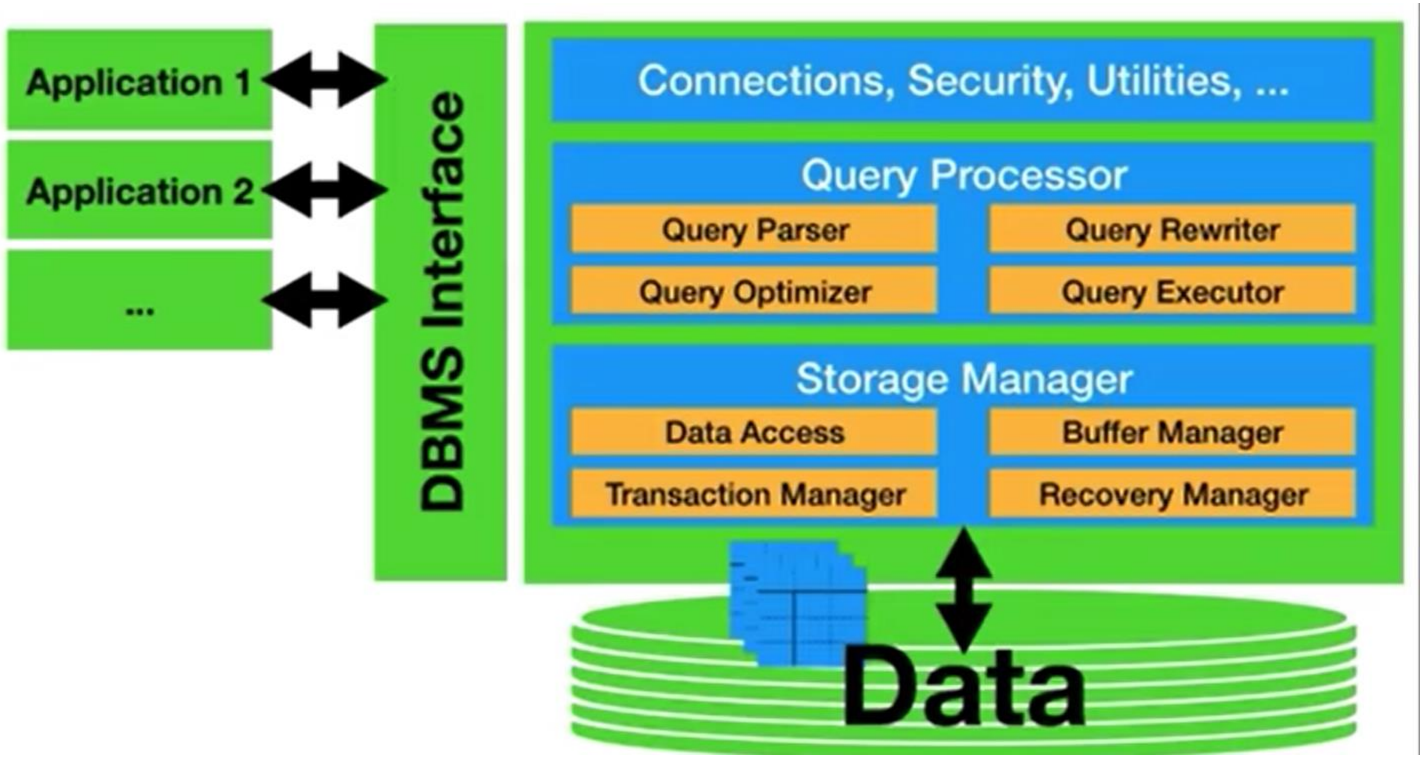
abc.py

xyz.py

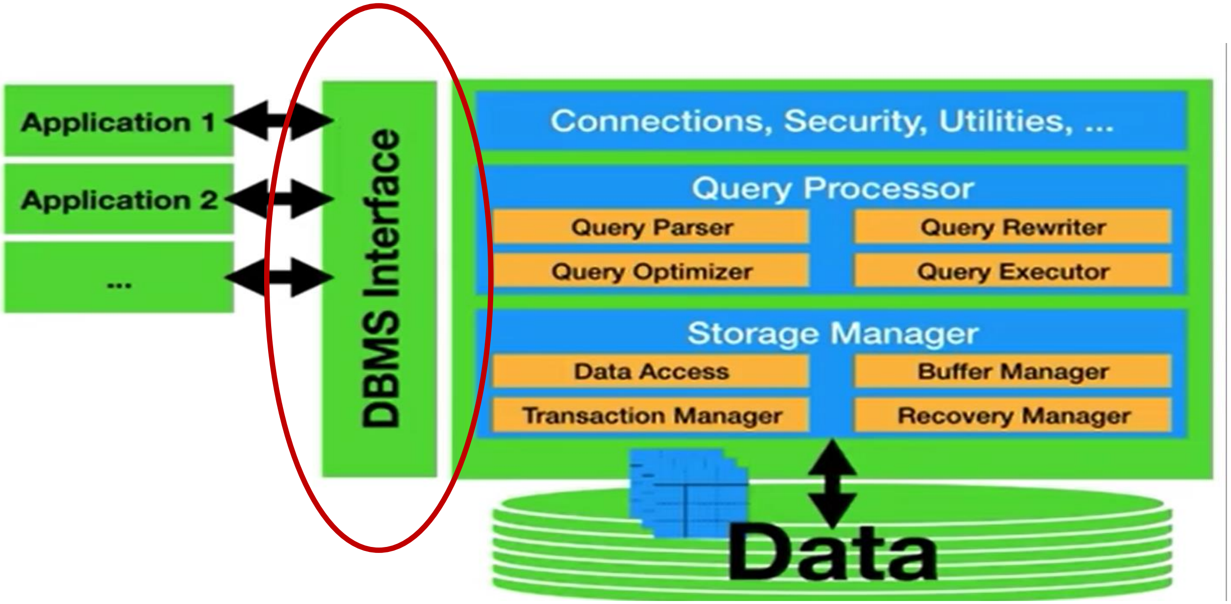
abc.csv

xyz.csv

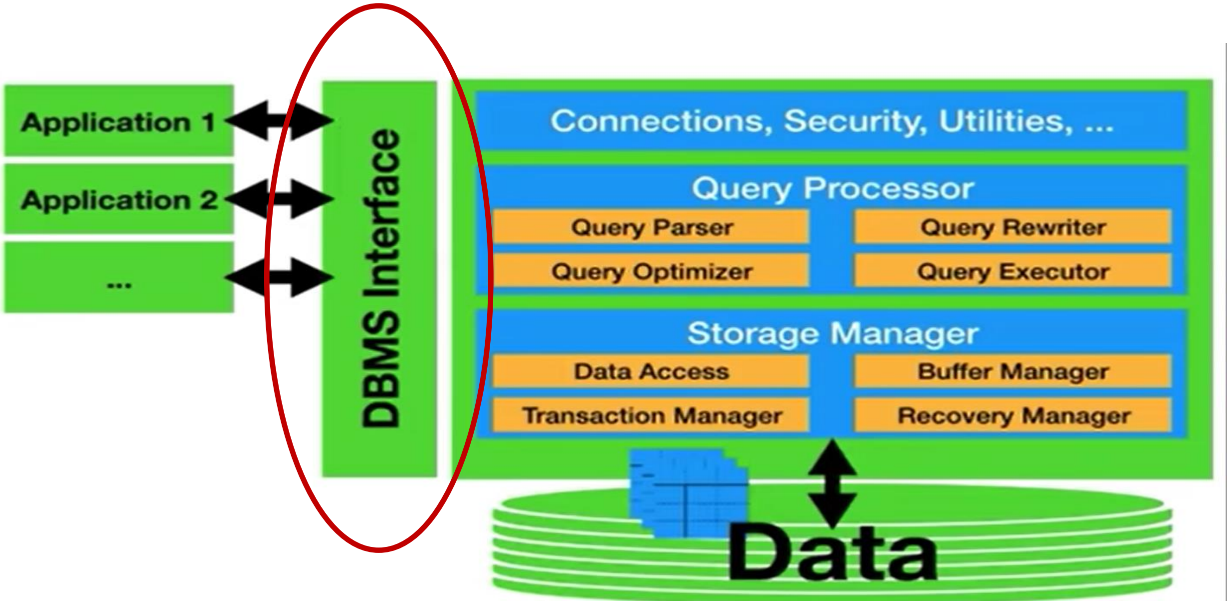
Database Management System (DBMS)



What should be the DBMS Interface?



What should be the DBMS Interface?

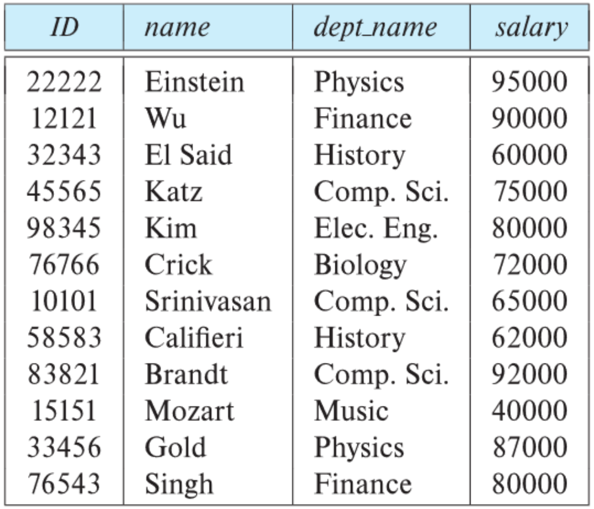


- Data Model
  - A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

The "Relational" Model ← "TABLE - based Data Model"

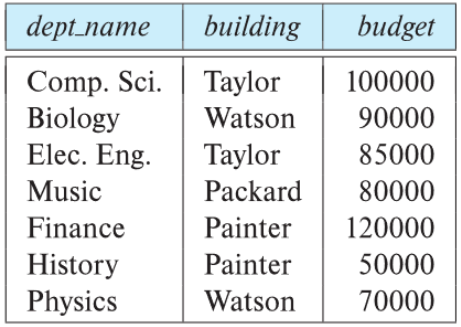
- The relational model uses <u>a collection of tables</u> to represent both data and the relationships among those data.

- Its conceptual simplicity has led to its widespread adoption; a vast majority of database products are based on the relational model.



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

(a) The *instructor* table

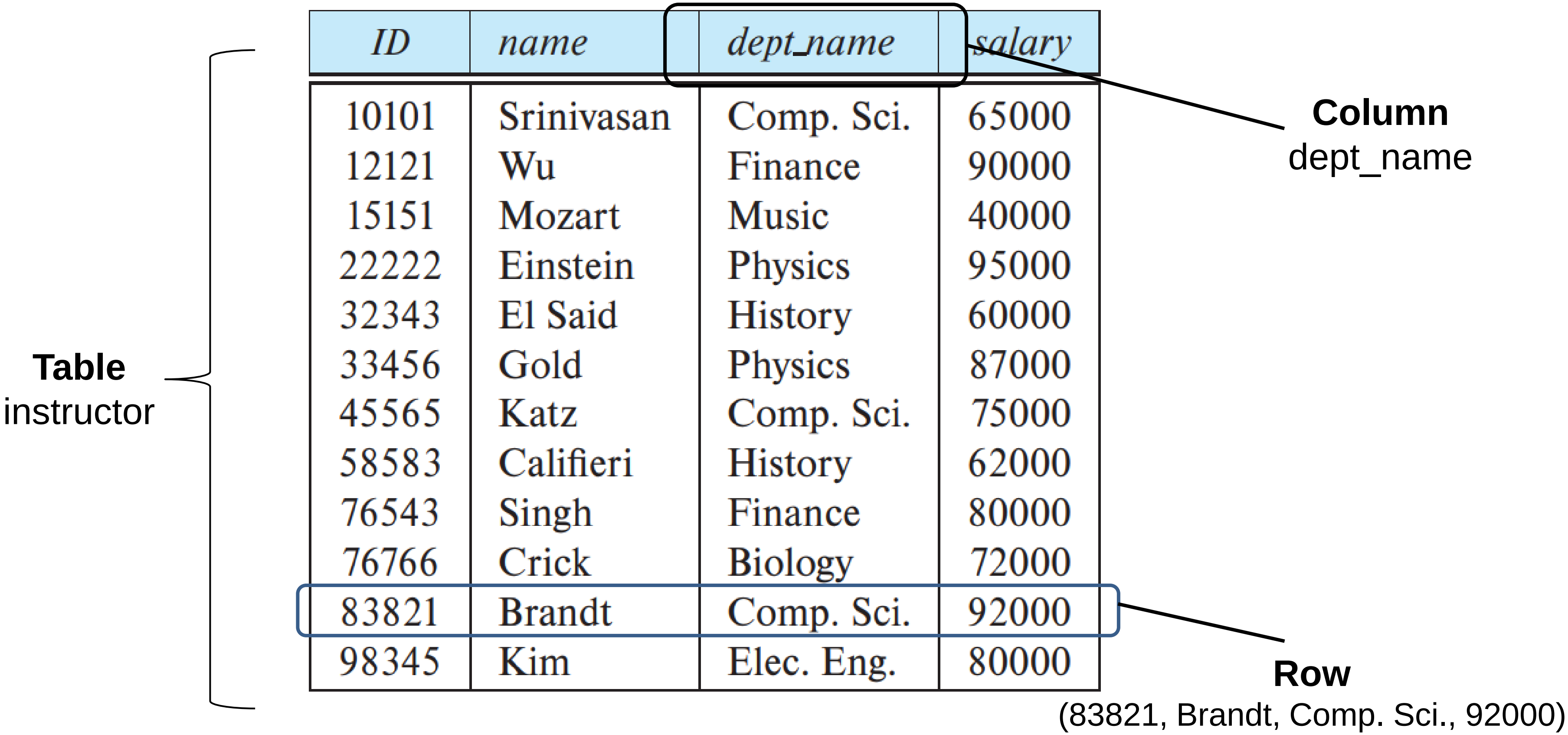


dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

The Relational Model

- The relational model uses a collection of tables to represent both data and the relationships among those data.



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

Table
instructor

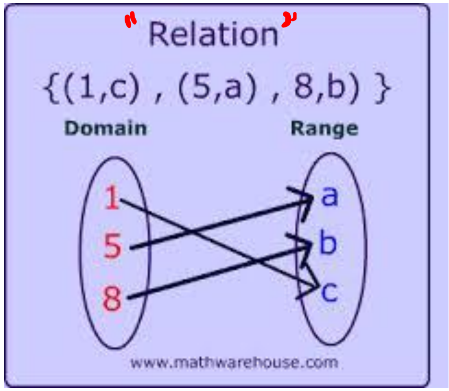
Column
dept_name

Row
(83821, Brandt, Comp. Sci., 92000)

Figure 2.1 The *instructor* relation.

Why is It Called the "Relational" Model?

- Let's talk some math!

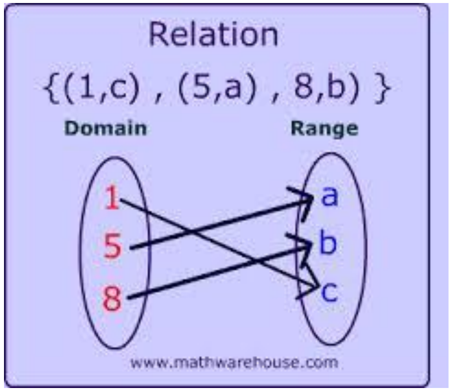


- The relation between two sets is a collection of pairs (2-tuples) containing one object from each set.

Why is It Called the "Relational" Model?

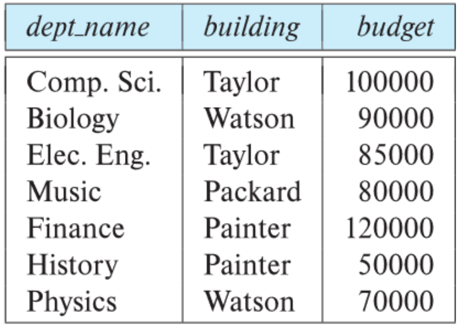
$\{(1,a,A),(5,b,B)\ldots\ldots\}$

- Let's talk some math!



- The relation between two sets is a set of pairs (2-tuples) containing one object from each set.
- The relation between three sets is a set of 3-tuples containing one object from each set.
- The relation between n sets is a set of n-tuples containing one object from each set.

Why is It Called the "Relational" Model?



dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

dept_name = { Comp Sci, Biology, EE, Music, ... }

building : { Taylor, Watson, Packard, Painter }

budget : { Set of All integer }

Relation on these 3 sets = { (Comp. Sci, Taylor, 100000), (Biology, Watson, 9000

(EE, Taylor, 85000),

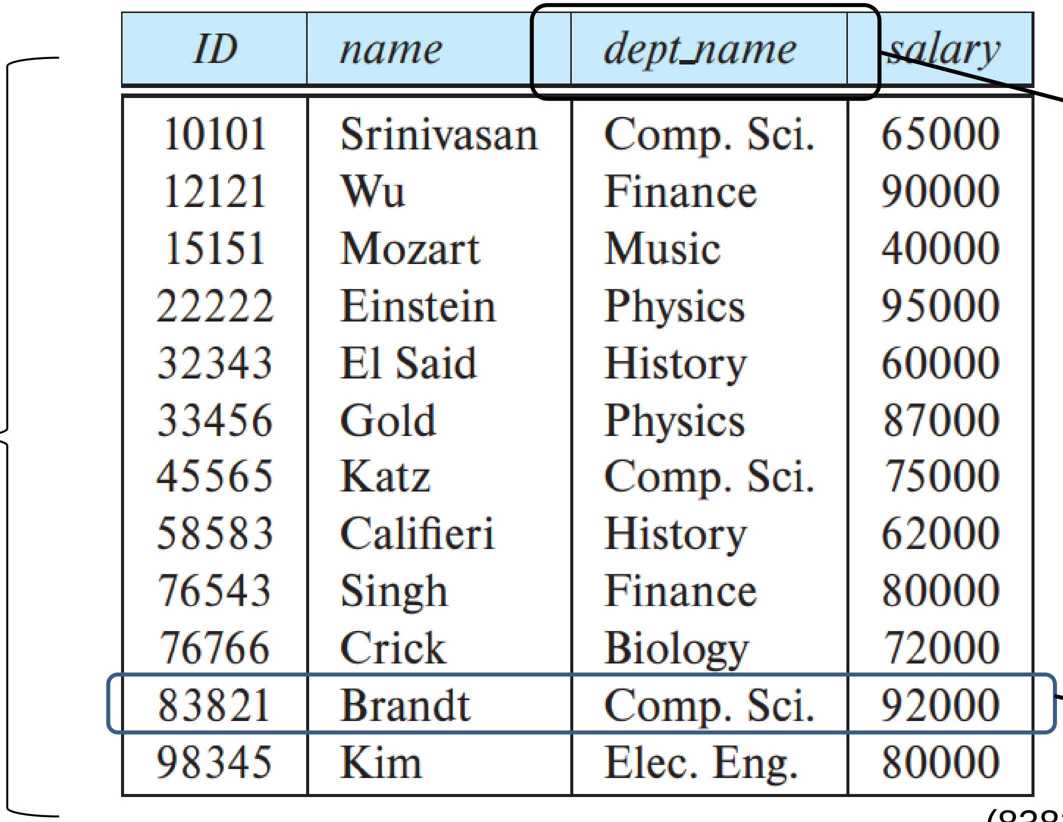
⋮

}

- The relation is a set of tuples.

The Relational Model

- The relational model uses a collection of tables to represent both data and the relationships among those data.



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

Table/ Relation
instructor

Column / Attribute
dept_name

Row / Tuple
(83821, Brandt, Comp. Sci., 92000)

Figure 2.1 The *instructor* relation.

Let’s Practice Our Terminology!

{ Biology, Comp. Sci, Elect. Eng. Finance, History, Music, Physics. MECH Educ. }

- Identify some attributes of the *course* relation?

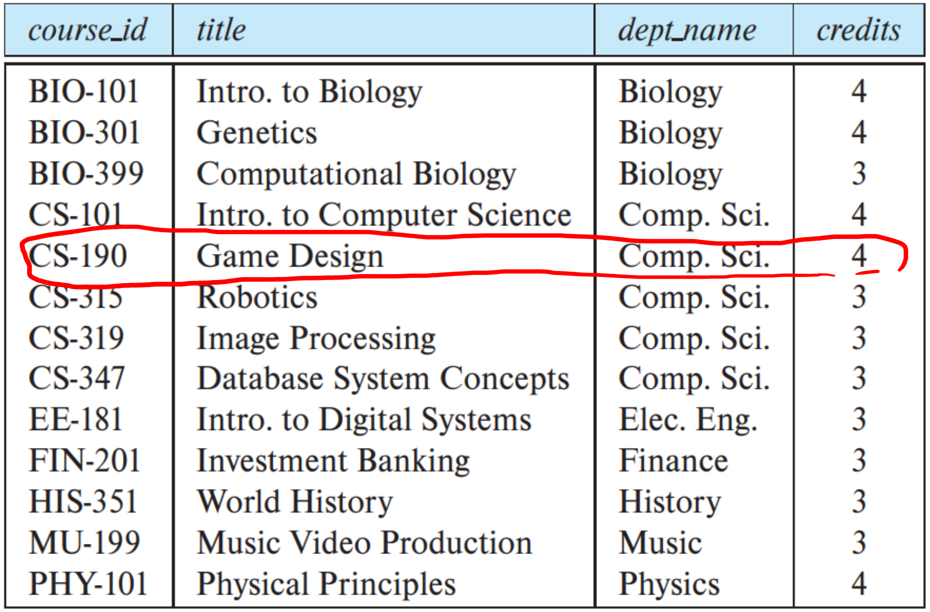
course_id, title

- Identify any tuple in the *course* relation?

(CS-190, GameDesign, Comp. Sci, 4)

- Are there any attributes that have a unique value in each tuple of this relation?

course_id, title



course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Figure 2.2 The *course* relation.

Properties of Attributes

Comp. ID	Company Name	Office Location
1	ENGRO	Lahore, Karachi, Islamabad
2	UNILEVER	Lahore, Karachi.
- -	- -	- -

Comp ID	Comp. Name	Office Location
1	ENGRO	Lahore
1	ENGRO	Karachi
1	ENGRO	Islamabad
2	Unilever	Lahore
2	Unilever	Karachi.

- For each attribute of a relation, there is a set of permitted values, called the **domain** of that attribute.

- For all relations, domains of all attributes must be **atomic**.

- A domain is atomic if elements of the domain are considered to be indivisible units.

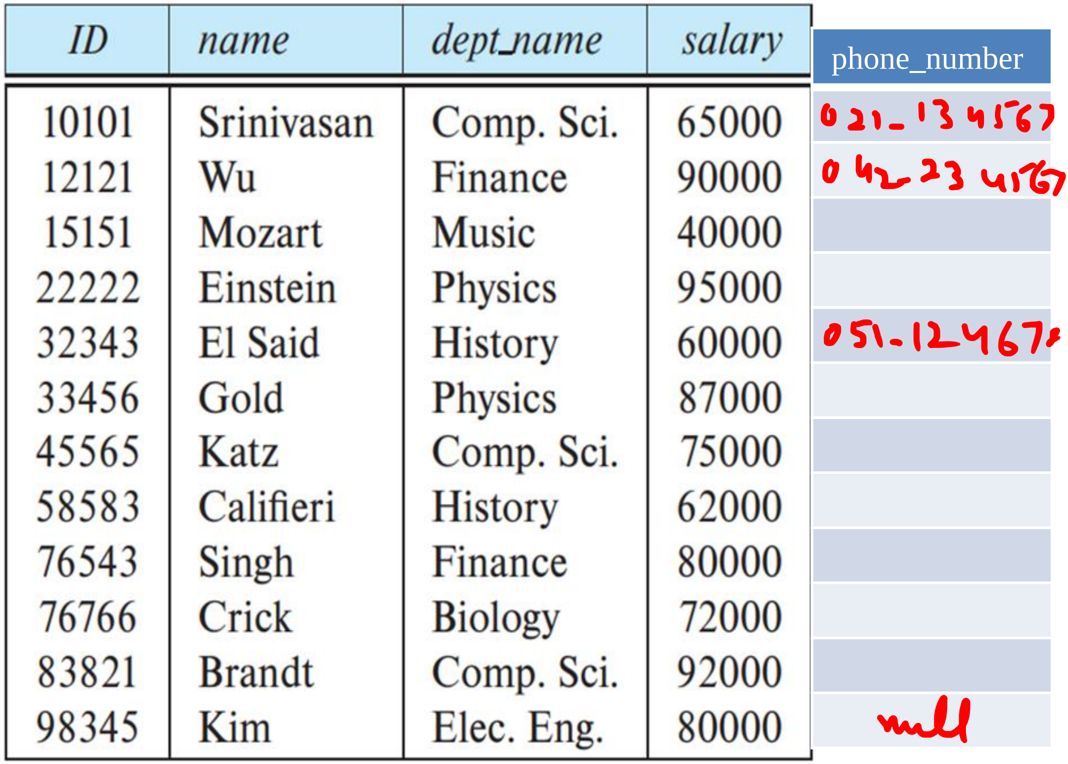
- The **null** value is a special value that signifies that the value is unknown or does not exist.

Properties of Attributes: Example

- Suppose that we add an attribute "phone_number" to the *instructor* relation.

- Is this attribute atomic?

- Can the attribute have a **null** value?



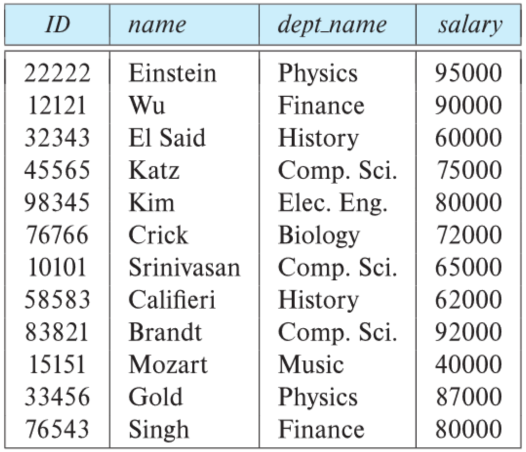
ID	*name*	*dept_name*	*salary*	phone_number
10101	Srinivasan	Comp. Sci.	65000	021-134567
12121	Wu	Finance	90000	042-234167
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	051-12467
33456	Gold	Physics	87000	
45565	Katz	Comp. Sci.	75000	
58583	Califieri	History	62000	
76543	Singh	Finance	80000	
76766	Crick	Biology	72000	
83821	Brandt	Comp. Sci.	92000	
98345	Kim	Elec. Eng.	80000	null

Database: Schema vs Instance

- Database Schema
  - Logical design of the database

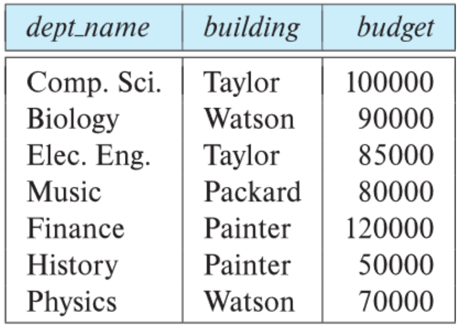
department (dept_name, building, budget)
instructor (ID, name, dept_name, salary)

- Database Instance
  - A snapshot of the data in the database at a give instant in time



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

(a) The *instructor* table



dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

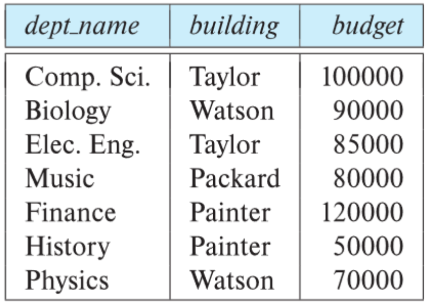
(b) The *department* table

Relation: Schema vs Instance

- Relation Schema
  - Consists of a list of attributes and their corresponding domains

department (dept_name, building, budget)

- Relation Instance
  - A snapshot of the tuples in a relation at a give instant in time

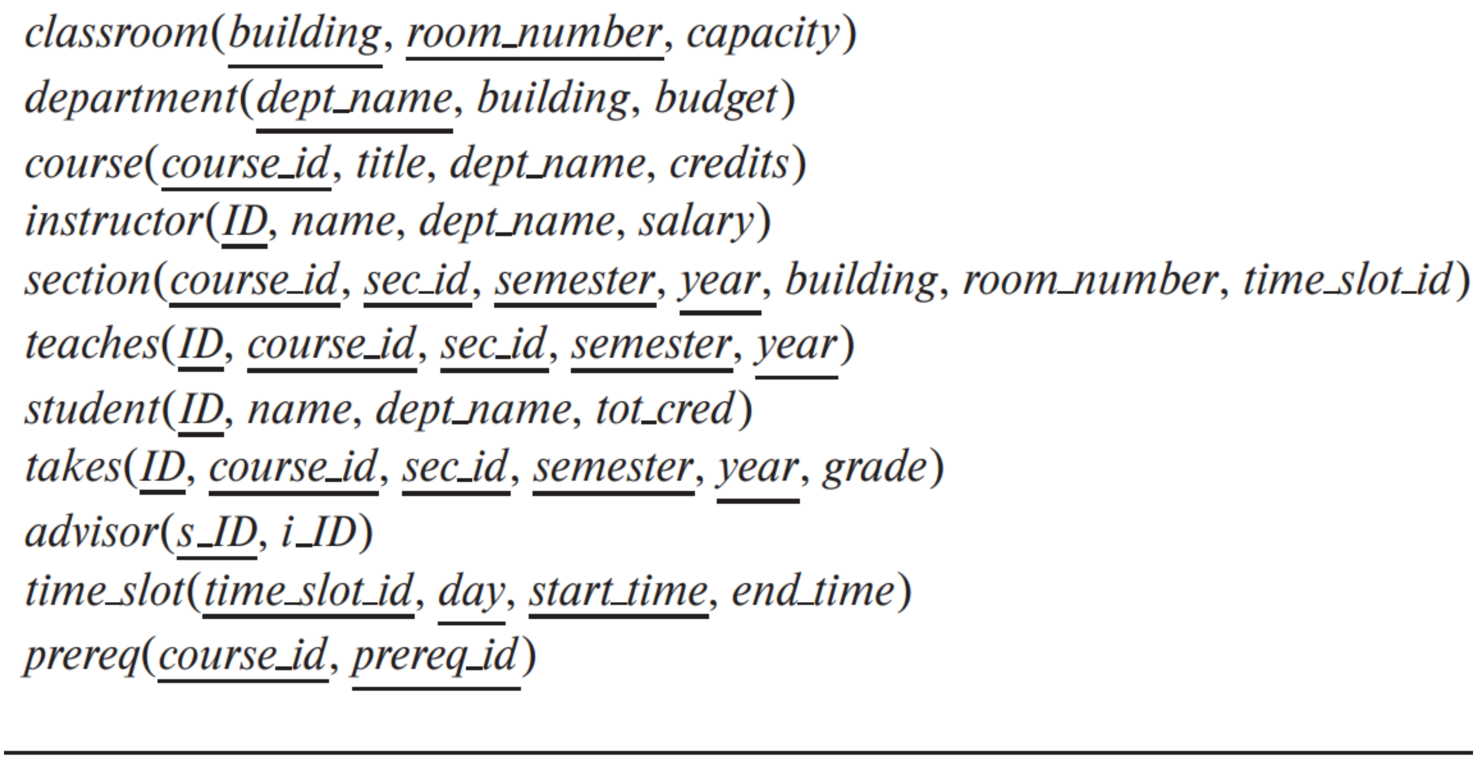


dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

- We often use the same name (such as *department*) to refer to both the schema and the instance. However, when required, we can differentiate:
  - “The *department* schema”
  - “An instance of the *department* relation”

Database Schema: Example



$\mathit{classroom}(\underline{\mathit{building}}, \underline{\mathit{room\_number}}, \mathit{capacity})$

$\mathit{department}(\underline{\mathit{dept\_name}}, \mathit{building}, \mathit{budget})$

$\mathit{course}(\underline{\mathit{course\_id}}, \mathit{title}, \mathit{dept\_name}, \mathit{credits})$

$\mathit{instructor}(\underline{\mathit{ID}}, \mathit{name}, \mathit{dept\_name}, \mathit{salary})$

$\mathit{section}(\underline{\mathit{course\_id}}, \underline{\mathit{sec\_id}}, \underline{\mathit{semester}}, \underline{\mathit{year}}, \mathit{building}, \mathit{room\_number}, \mathit{time\_slot\_id})$

$\mathit{teaches}(\underline{\mathit{ID}}, \underline{\mathit{course\_id}}, \underline{\mathit{sec\_id}}, \underline{\mathit{semester}}, \underline{\mathit{year}})$

$\mathit{student}(\underline{\mathit{ID}}, \mathit{name}, \mathit{dept\_name}, \mathit{tot\_cred})$

$\mathit{takes}(\underline{\mathit{ID}}, \underline{\mathit{course\_id}}, \underline{\mathit{sec\_id}}, \underline{\mathit{semester}}, \underline{\mathit{year}}, \mathit{grade})$

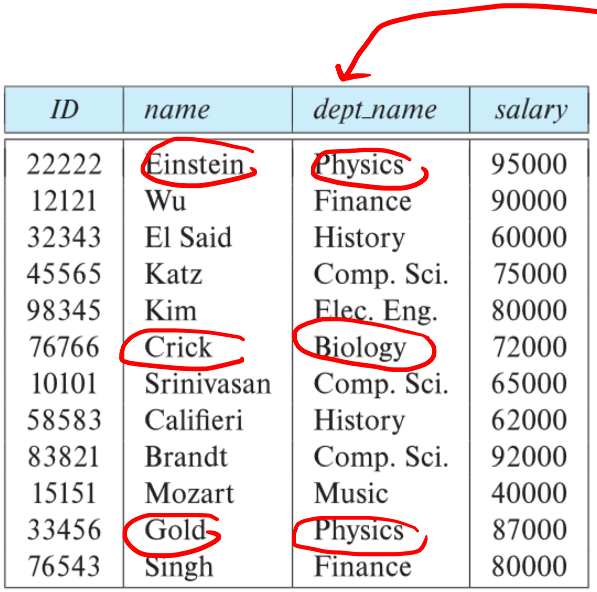
$\mathit{advisor}(\underline{\mathit{s\_ID}}, \mathit{i\_ID})$

$\mathit{time\_slot}(\underline{\mathit{time\_slot\_id}}, \underline{\mathit{day}}, \underline{\mathit{start\_time}}, \mathit{end\_time})$

$\mathit{prereq}(\underline{\mathit{course\_id}}, \underline{\mathit{prereq\_id}})$

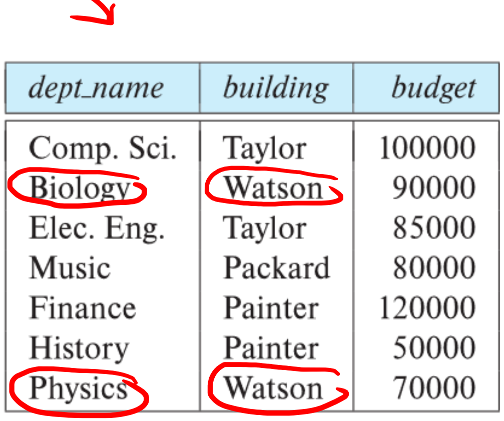
Figure 2.8 Schema of the university database.

Example of a Query



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

(a) The *instructor* table



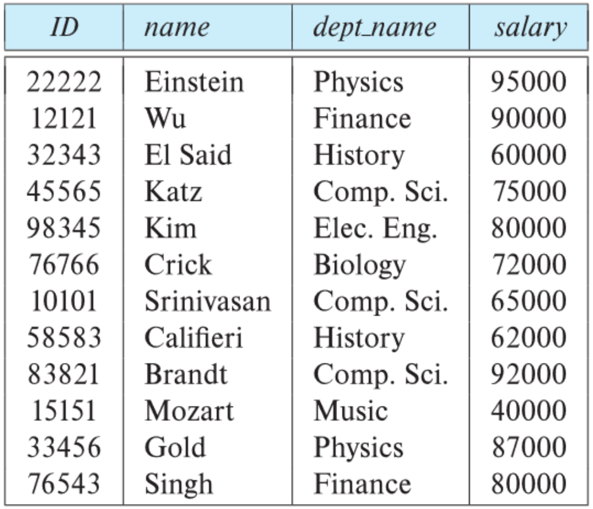
dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

- Find the names of all instructors who work in Watson building

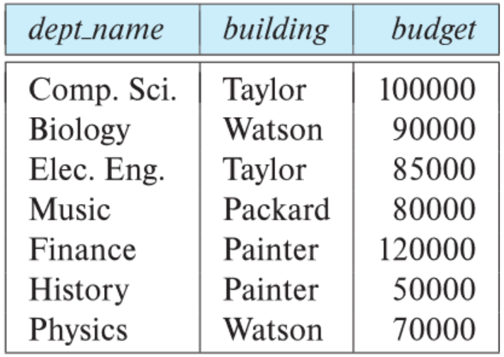
Einstein, Crick, Gold

Example of a Query: Role of Link between Relations



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

(a) The *instructor* table



dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

- Find the names of all instructors who work in Watson building
  - Consider the department schema and the instructor schema
  - Observe that the attribute ***dept_name*** is duplicated in both.
  - This duplication is useful in answering queries which involve multiple relations.

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following version of the ***department*** relation:

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000
CS	Dr. Abdul Samad	100000

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following versions of the ***department*** relation:

dept-name	**program-director**	**budget**
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000
CS	Dr. Abdul Samad	100000

- No two tuples in a relation are allowed to have exactly the same value for all attributes.

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following version of the ***department*** relation:

dept-name	**program-director**	**budget**
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000
CS	Dr. Waqar	100000

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following version of the ***department*** relation:

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000
CS	Dr. Waqar	100000

- Values for some attributes cannot be repeated in a relation.

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following versions of the ***department*** and ***instructor*** relations:

department

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000

instructor

ID	name	dept-name
2033	Dr. Abdul Samad	CS
2071	Dr. Farhan	ECE
3045	Dr. Pervez	Physics
3067	Dr. Waqar Saleem	ISciM

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following relations:

department

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000

instructor

ID	name	dept-name
2033	Dr. Abdul Samad	CS
2071	Dr. Farhan	ECE
3045	Dr. Pervez	Physics
3067	Dr. Waqar Saleem	ISciM

- Some attributes can only be assigned values that are present in another relation.

Constraints on Tuples in a Relation: What is the Source?

- Any relational model (that you develop) is trying to model a real-world enterprise/business.
  - For instance, in the previous slides, the relational model was an attempt at modeling the DSSE at Habib University.

- Constrains on the tuples in a relational model (such as the examples shown in previous slides) are dictated by the rules/constraints of the real-world enterprise/business being modeled.

DSSE
@
HU

department(dept_name, prog-director, budget)

instructor (ID, name, dept_name)

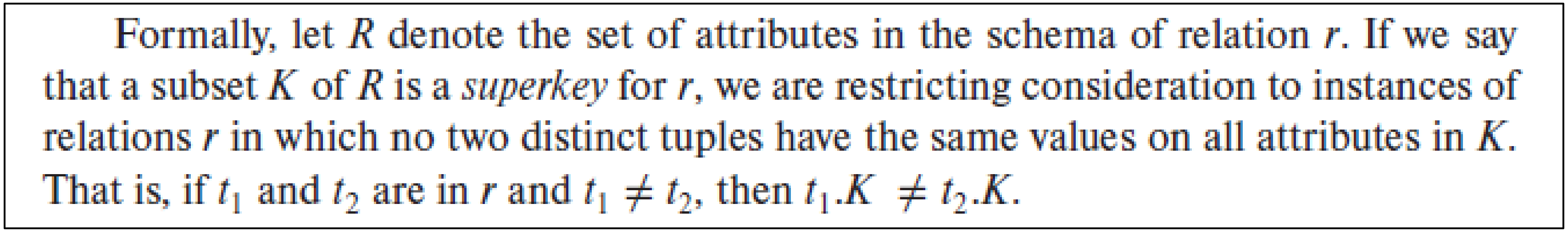
Constraints on Tuples in a Relation: How are these Constraints Expressed/Specified?

- Through “Keys”

- Through the following types of “Keys”, we specify how tuples within a given relation must be different from one another.
  - Superkey
  - Candidate Key
  - Primary Key
  - Foreign Key

Keys: Superkey

- A **superkey** K is a set of one or more attributes such that no two distinct tuples can have the same values on all attributes in the set K.



Formally, let R denote the set of attributes in the schema of relation r. If we say that a subset K of R is a *superkey* for r, we are restricting consideration to instances of relations r in which no two distinct tuples have the same values on all attributes in K. That is, if t_1 and t_2 are in r and $t_1 \neq t_2$, then $t_1.K \neq t_2.K$.

- **{ID}?**

  Super key

- **{Name}** ?

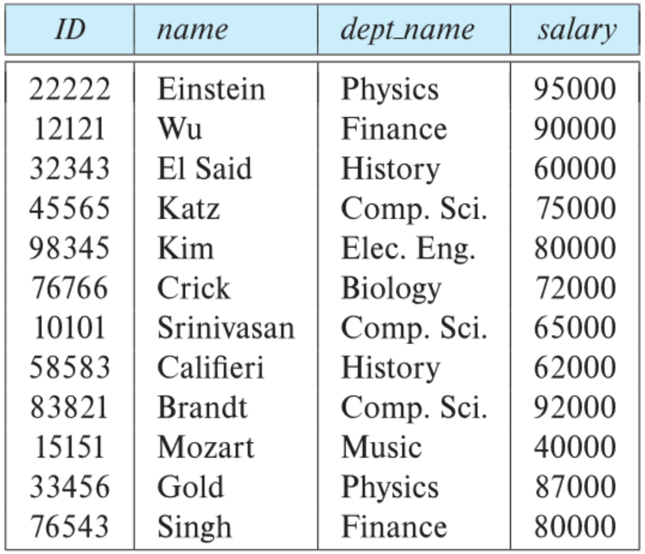
  Not a Superkey

- **{ID, Name}**?

  Superkey

- **{name, dept_name, salary}**?

  Not a Superkey

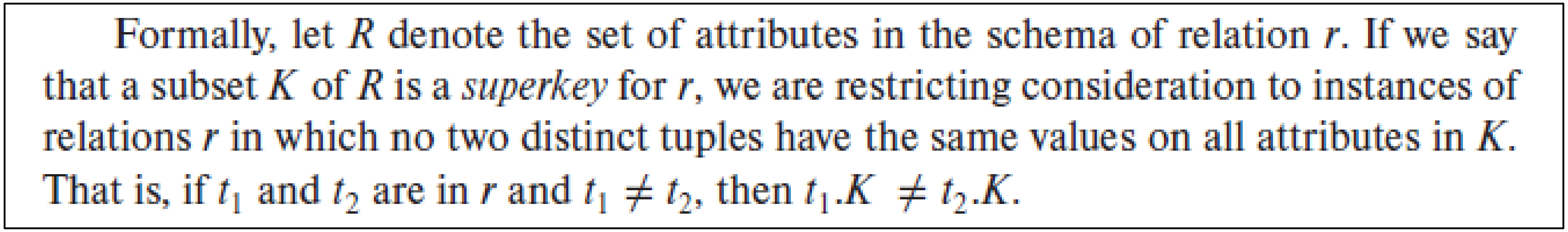


ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

(a) The *instructor* table

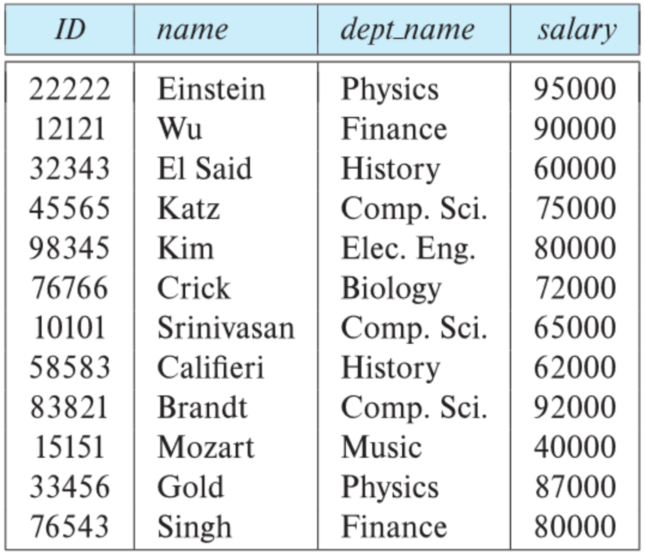
Keys: Superkey

- A **superkey** K is a set of one or more attributes such that no two distinct tuples can have the same values on all attributes in the set K.



Formally, let R denote the set of attributes in the schema of relation r. If we say that a subset K of R is a *superkey* for r, we are restricting consideration to instances of relations r in which no two distinct tuples have the same values on all attributes in K. That is, if t_1 and t_2 are in r and $t_1 \neq t_2$, then $t_1.K \neq t_2.K$.

- **{ID}?**
  - is a Superkey: enough to identify a tuple uniquely
- **{Name}** ?
  - is not a Superkey: two instructor can have the same name
- **{ID, Name}**?
  - is a superkey: enough to identify a tuple uniquely
- **{name, dept_name, salary}**?
  - Not a superkey. Possible for two instructors to have the same values of name, dept_name, salary

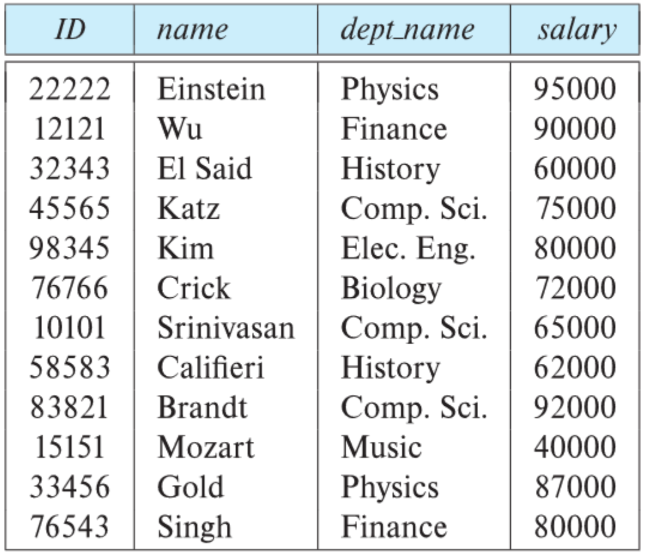


ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

(a) The *instructor* table

Keys: Candidate Keys

- A superkey may contain extraneous attributes.
- For example, each of the following is a superkey:
  - ***{ID}***
  - ***{ID, name}***
  - ***{ID, name, dept_name}***
- In general, if *K* is a superkey, then so is any superset of *K*.
- We are often interested in superkeys for which no proper subset is a superkey.
- Such minimal superkeys are called **candidate keys**.
  - Only ***{ID}*** is the candidate key.



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

(a) The *instructor* table

Keys: Candidate Keys

- Consider the following relation schema:

student (name, date_of_birth, major, student_id, address)

- Candidate Keys?

{student_id} ✓

{name, address}

Keys: Candidate Keys

- Consider the following relation schema:

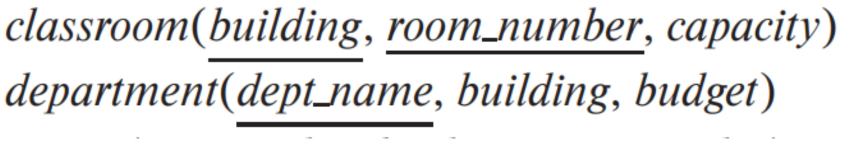
student (name, date-of-birth, major, student-id, address)

- Candidate Keys
  - {student-id}
  - {name, address}

- It is possible that several distinct sets of attributes could serve as a candidate key.

Keys: Primary Key

- Primary Key
  - One of the candidate keys that is chosen as the principle means of identifying tuples within a relation
  - The choice is made by the database designer
  - Notation:



$\textit{classroom}(\underline{\textit{building}}, \underline{\textit{room\_number}}, \textit{capacity})$

$\textit{department}(\underline{\textit{dept\_name}}, \textit{building}, \textit{budget})$

- Equivalent Term: Primary Key Constraint
  - The designation of a key represents a constraint in the real-world enterprise being modeled through the relational model.
  - Therefore, “primary keys” are also referred as “primary key constraints”.

Keys as a Representation of Constraints in the Real World

- Consider the following alternative relation schemas:

1) student (student-id , major, year-of-entry, date-of-birth, address)

2) student (student-id , major, year-of-entry, date-of-birth, address)

3) student (student-id , major, year-of-entry, date-of-birth, address)

- What does each option tell you about the Student ID conventions being used by the University for which the schema is being developed?

(1)

~~20~~

ID
2003-EE-205

(2)

ID	Major
2003-205	EE
2003-205	CE

(3)

ID	Major	Year of Entry
205	EE	2003
205	EE	2004

Constraints on Contents/Tuples in a Relation: Examples

- Consider the following relations:

department

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000

?

instructor

ID	name	dept-name
2033	Dr. Abdul Samad	CS
2071	Dr. Farhan	ECE
3045	Dr. Pervez	Physics
3067	Dr. Waqar Saleem	ISciM

- Some attributes can only be assigned values that are present in another relation.

Keys: Foreign Key

- Consider the following relations:

department

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000

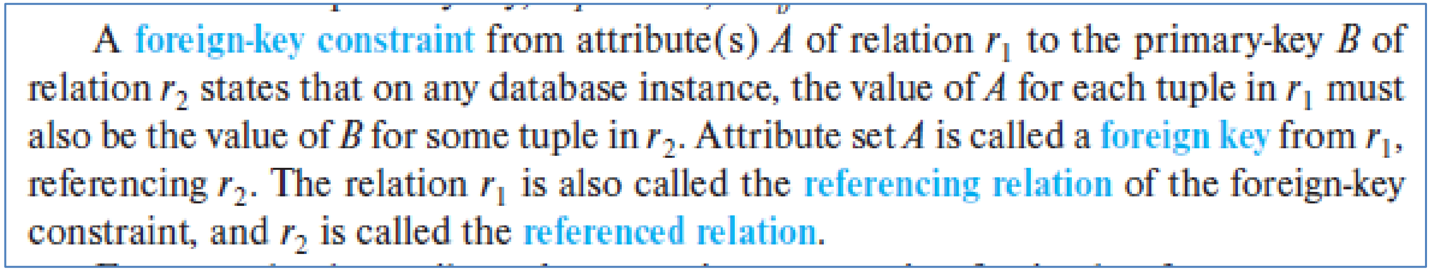
instructor

ID	name	dept-name
2033	Dr. Abdul Samad	CS
2071	Dr. Farhan	ECE
3045	Dr. Pervez	Physics
3067	Dr. Waqar Saleem	ISciM

- Attribute *dept_name* in relation *instructor* can only be assigned values that are present as the values of attribute *dept_name* in relation *department*.
- ***dept_name*** is a **foreign key** from the relation ***instructor*** to the relation ***department***
- **Referencing relation**: *instructor* **Referenced relation**: *department*
- The referenced attributes (*dept_name*) must be the primary key in the referenced relation.

Keys: Foreign Key

- Formally:



A foreign-key constraint from attribute(s) A of relation r_1 to the primary-key B of relation r_2 states that on any database instance, the value of A for each tuple in r_1 must also be the value of B for some tuple in r_2. Attribute set A is called a foreign key from r_1, referencing r_2. The relation r_1 is also called the referencing relation of the foreign-key constraint, and r_2 is called the referenced relation.

department (dept-name, program-director, budget)
Instructor(ID, name, dept-name)

department

dept-name	program-director	budget
CS	Dr. Abdul Samad	100000
ECE	Dr. Farhan	85000
ISciM	Dr. Aeyaz	80000

instructor

ID	name	dept-name
2033	Dr. Abdul Samad	CS
2071	Dr. Farhan	ECE
3045	Dr. Pervez	Physics
3067	Dr. Waqar Saleem	ISciM

Keys: Exercise

Item (Item Name, Vendor Name, Price)
Customer (Customer Code, Customer Name, Address)
Vendor (Vendor Name, Contact Person, Address)
Order(Order No, Customer Code, Item Name, Vendor Name, Order Date)

- Identify Primary Keys?

Keys: Exercise

Item (Item Name, Vendor Name, Price)
Customer (Customer Code, Customer Name, Address)
Vendor (Vendor Name, Contact Person, Address)
Order(Order No, Customer Code, Item Name, Vendor Name, Order Date)

- Identify Foreign Keys?

{Customer Code} from Order to Customer

{Item Name, Vendor Name} from Order to Item

{Vendor Name} from Item to Vendor

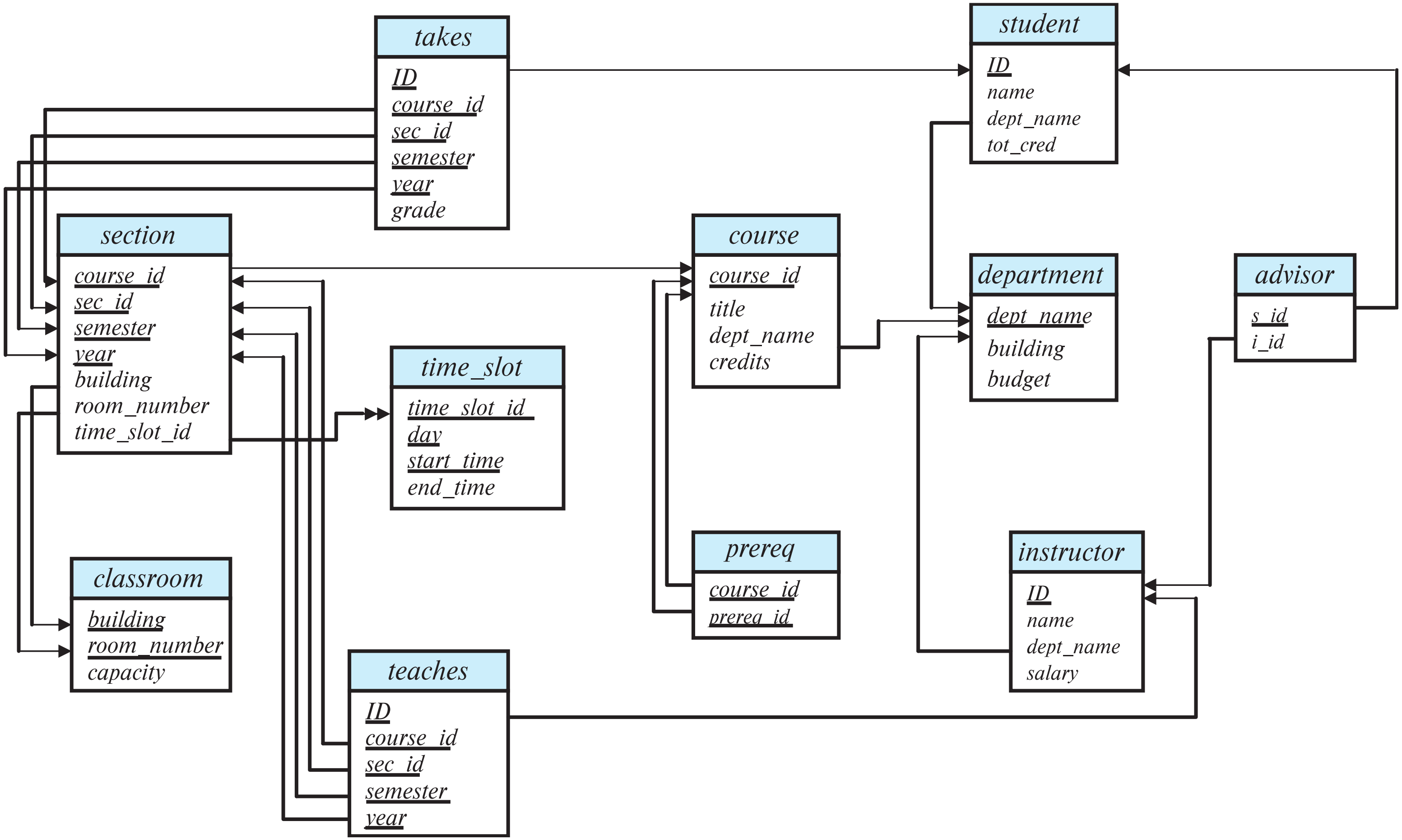
Keys: Exercise

Item (Item Name, Vendor Name, Price)
Customer (Customer Code, Customer Name, Address)
Vendor (Vendor Name, Contact Person, Address)
Order(Order No, Customer Code, Item Name, Vendor Name, Order Date)

- Identify Foreign Keys?
  - {Customer Code} is a foreign key from Order to Customer
  - {Item Name, Vendor Name} is a foreign key from Order to Item
  - {Vendor Name} is a foreign key from Item to Vendor

Schema Diagram

- Used to depict a database schema along with primary key and foreign key dependencies



takes
ID
course_id
sec_id
semester
year
grade

student
ID
name
dept_name
tot_cred

section
course_id
sec_id
semester
year
building
room_number
time_slot_id

course
course_id
title
dept_name
credits

department
dept_name
building
budget

advisor
s_id
i_id

time_slot
time_slot_id
day
start_time
end_time

prereq
course_id
prereq_id

instructor
ID
name
dept_name
salary

classroom
building
room_number
capacity

teaches
ID
course_id
sec_id
semester
year

Schema Design 101

- Identify Tables
- Identify Columns/Attributes associated with each table
- Identify Primary Keys
- Identify relationships among tables through Foreign Keys

Practice Exercise 1

works

~~Em~~ person-name	company	Benefits
Usman	HU	
Usman	HU	

- For the following relational schema
  - Identify all primary keys (underline)
  - Identify all foreign keys (use arrows)

employee (person_name, street, city)
works (person_name, company_name, salary)
company (company_name, city)

employee (CNIC, person-name, street, city)

works (CNIC , person-name, company-id, ~~co~~ company-name, salary)

company (companyID , company-name, city)

Practice Exercise 2

- For the following relational schema
  - Identify all primary keys (underline)
  - Identify all foreign keys (use arrows)

$\textit{branch}(\underline{\textit{branch\_name}}, \textit{branch\_city}, \textit{assets})$

$\textit{customer}\ (\underline{\textit{ID}}, \textit{customer\_name}, \textit{customer\_street}, \textit{customer\_city})$

$\textit{loan}\ (\underline{\textit{loan\_number}}, \boxed{\textit{branch\_name},}\ \textit{amount})$

$\textit{borrower}\ (\underline{\textit{ID}}, \textit{loan\_number})$

$\textit{account}\ (\textit{account\_number}, \textit{branch\_name}, \textit{balance})$

$\textit{depositor}\ (\underline{\textit{ID}}, \textit{account\_number})$

Practice Exercise 3

- You have been asked to create a database schema for a hospital. For your help, the following relations have been identified in the schema:
  - physician
  - department
  - affiliated_with
  - procedure
  - prescribes
  - room
  - stay
  - undergoes
  - trained_in
  - patient
  - appointment
  - medication

- Complete the process of developing the schema