CSCB20
Introduction to Databases and Web Application

Week 1 - Relational Algebra

Dr. Purva R Gawde

Thanks to Dr. Anna Bretscher for the material in this set of slides

Topics covered this week

- Quick Review of terminology
- Relational Model Continued
  - Relational diagrams
  - Relational operations
  - Relational algebra

Review of last Week

Example of Relational Database - University

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

The *course* relation

ID	course_id	sec_id	semester	year
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

The *teaches* 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

The *instructor* relation

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

The *department* relation

Example of a 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 Schema: instructor(ID, name, dept_name, salary)`

Terminology - Schema

- What is database schema?
  - A logical design of the database
- What is database instance?
  - A snapshot of the data in the database at a given instant in time
- What is a Relation Schema?
  - A list of attributes and their corresponding domains.
- What is the domain of an attribute?
  - A set of permitted values

Terminology - Relational Model

- What is a Data Model?
  - a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

- What is a Relational Model?
  - The relational model is a data model that uses a collection of tables to represent both data and the relationships among those data.

Terminology - Key

- What is a **key**?
  - Keys provide a way to specify how tuples within a given relation are distinguished
- What is a **superkey**?
  - A set of one or more attributes that uniquely identify a tuple in the relation.
- What is a **candidate key**?
  - A minimal super key.
- What is a **primary key**?
  - A candidate key chosen to distinguish between tuples.

Superkey - Example

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

The *instructor* relation

- Relational schema:
    - `instructor(ID, name, dept_name, salary)`
- Superkeys for relation instructor:
    - `{ID}, {name, dept_name}, {ID, name}`

Candidate Key - Example

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

The *instructor* relation

- Relation:
  - `instructor(ID, name, dept_name, salary)`
- Candidate keys for relation instructor:
  - `{ID}, {name, dept_name}`

Primary Key - Example

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

The *instructor* relation

- Relation:
  - `instructor(ID, name, dept_name, salary)`
- Candidate keys for relation instructor:
  - `{ID}, {name, dept_name}`
- Primary key for relation instructor:
  - `{ID}`
- Relational schema with primary key:
  - instructor(<u>ID</u>, name, dept_name, salary)

Foreign Keys

For establishing relationships and Referential Integrity

Relationship Between Tables

- A table may be related to other tables
  - Ex: An instructor teaches subjects
  - And subject is taught by an instructor during the course of various semesters

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

The *instructor* relation

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

The *teaches* relation

Foreign Key

- To establish relationships, we need to implement **a foreign key**
- A foreign key is a primary key from one table that is placed into another table.
- The key is called a foreign key in the table that receives the key

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

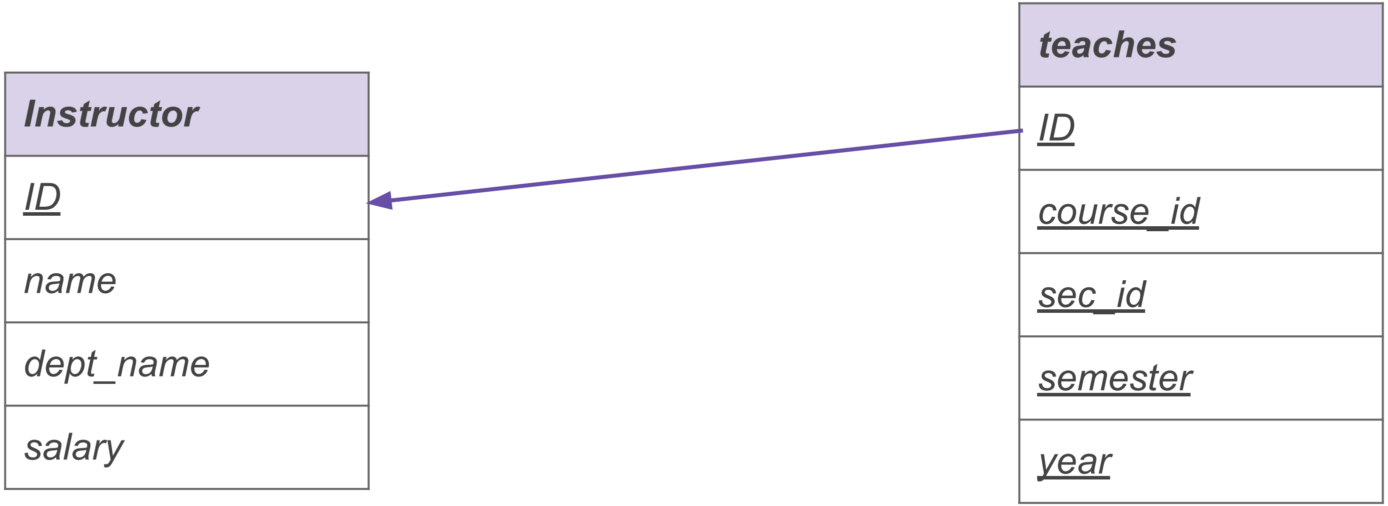
instructor(ID, name, dept_name, salary)

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

teaches(ID, course_id, sec_id, semester, year)

Foreign Key

- To establish relationships, we need to implement a foreign key
- A foreign key is a primary key from one table that is placed into another table.
- The key is called a foreign key in the table that receives the key



`instructor(ID, name, dept_name, salary)`   `teaches(ID, course_id, sec_id, semester, year)`

Foreign Keys

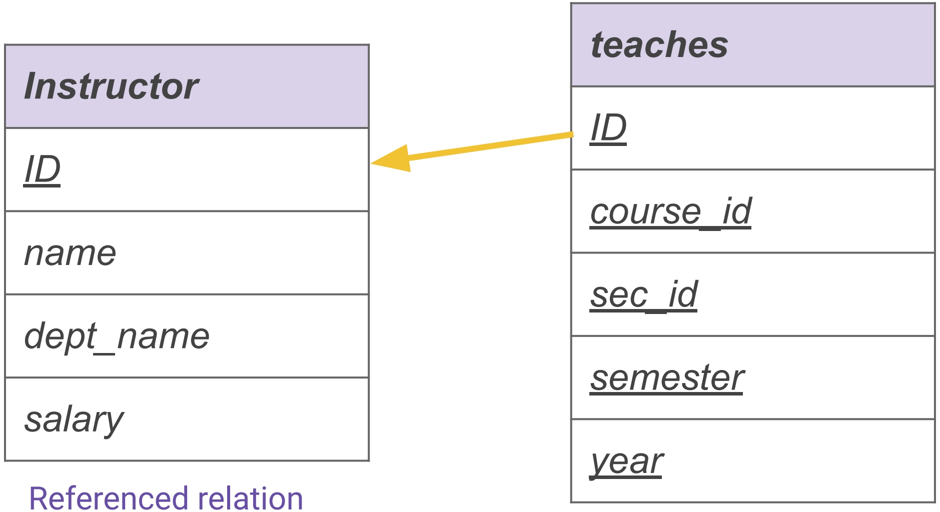
- A set of attributes in a relation (table) that is a primary key in another relation.
  - `instructor(ID, name, dept_name, salary)`
  - `department(dept_name, building, budget)`
  - `teaches(ID, course_id, sec_id, semester, year)`
- The primary keys are underlined.

- **What are the foreign keys for this set of relations?**
  - *dept_name* in instructor
  - *ID* in teaches

Foreign Keys

- `instructor(ID, name, dept_name, salary)`
- `department(dept_name, building, budget)`
- `teaches(ID, course_id, sec_id, semester, year)`

- `ID` from `teaches` references `instructor`.
- `teaches` is the referencing relation.
- `instructor` is the referenced relation.



Referenced relation

Referencing relation

Referential Integrity

- **Referential Integrity**
  Referential integrity states that every value of a foreign key **must** match a value of an existing primary key

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

`instructor(ID, name, dept_name, salary)`

ID	course_id	sec_id	semester	year
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

`teaches(ID, course_id, sec_id, semester, year)`

Referential Integrity

- **Also known as: Foreign Key Constraint**
  A foreign key value in one relation must appear in the referenced relation.

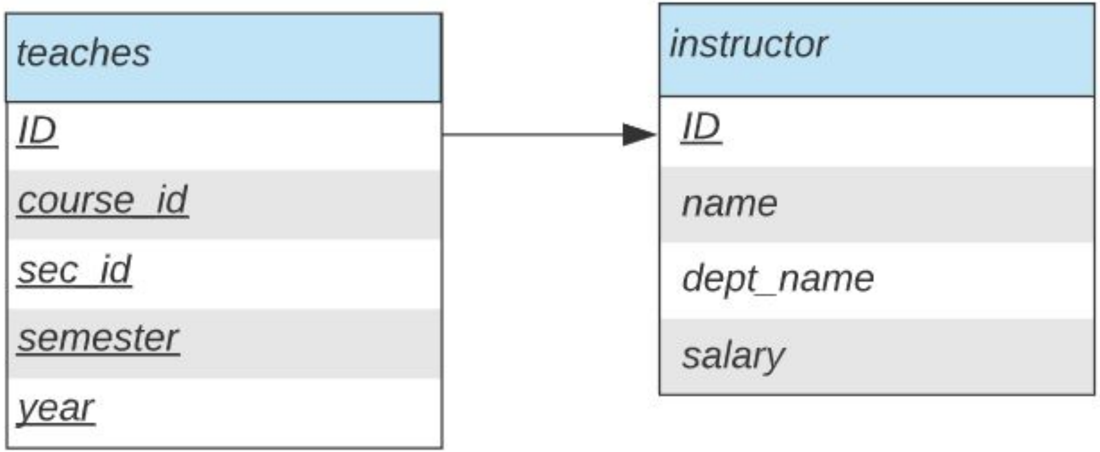
- **Example:**
  `teaches(ID, course_id, sec_id, semester, year)`
  `section(course_id, sec_id, semester, year, building, room_number, time_slot_id)`

- **What might be a foreign key constraint?**
  - course_id,sec_id, semester, year in teaches has a foreign key constraint on section.

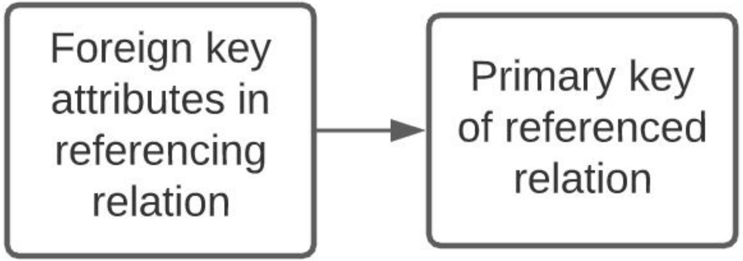
Schema Diagrams

Schema Diagrams

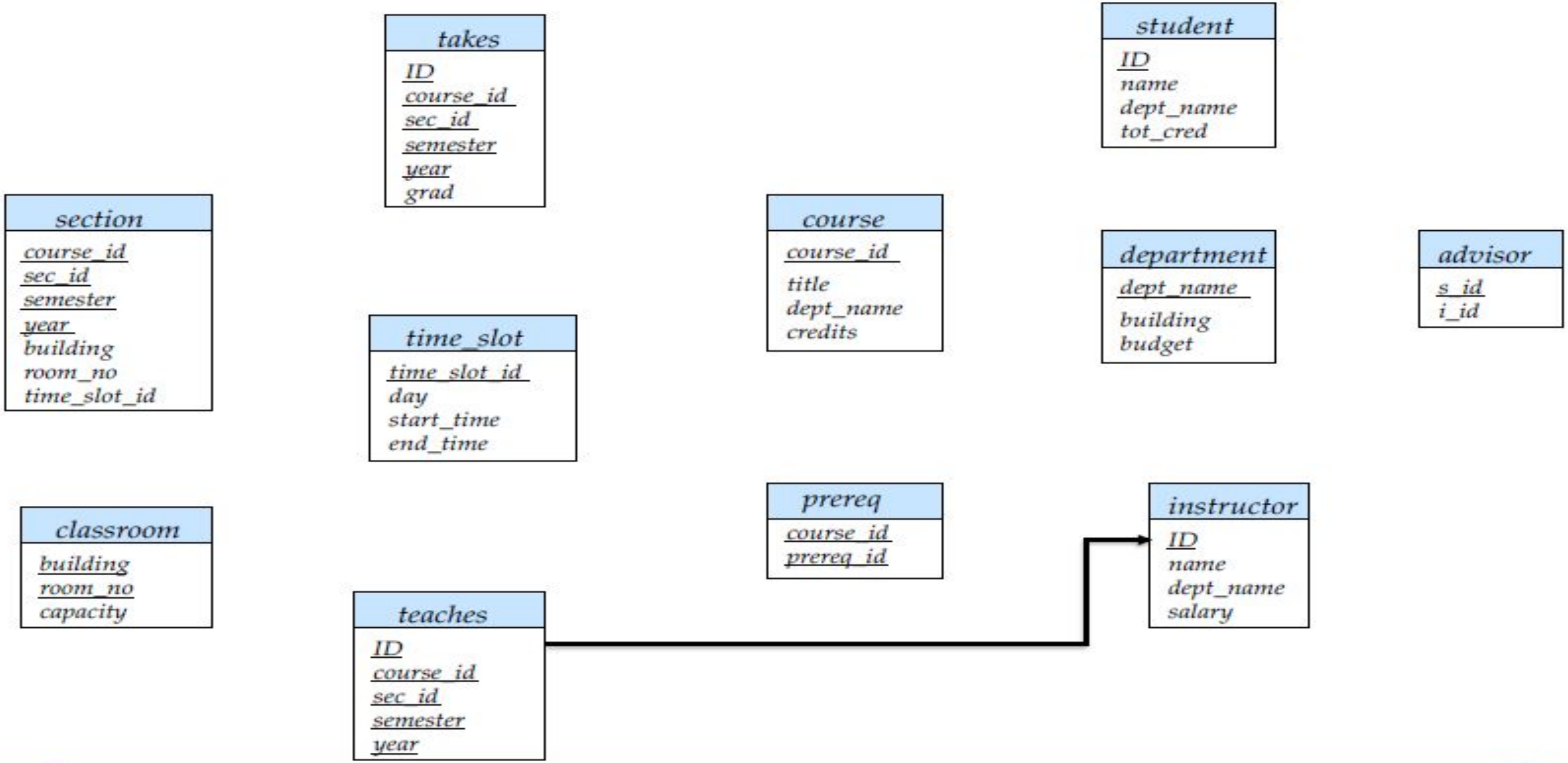
We can depict foreign key constraints and primary keys using a schema diagram.



The relation is in **light blue.**
Primary keys are <u>underlined</u>.



Add the Arrows



Relational Query Languages and Relational Algebra

Relational Query Languages

- We have a set of tables or relations.
- Now what? How do we get information from them?
  - We perform queries.
- How to perform those queries?
  - We need a Query Language.
- A query language is a
  - language in which a user requests information from database.
  - These languages are on a level higher than that of a standard programming language.
- Example of query languages: **Relational algebra**, Tuple relational calculus, Domain relational calculus

* We study the very widely used query language SQL in upcoming lectures

Relational Algebra

- Why learn Relational Algebra?
  - Forms the theoretical basis of the SQL query language.
  - Consists of a set of operations that take one or two relations as input and produce a new relation as their result.
- Unary operations: operate on one relation.
  - select, project, and rename
- Binary operations: operate on pairs of relations
  - union, Cartesian product, intersection, and set difference

Simple Query format

Query languages provide a set of operations that can be applied to either a single relation or a pair of relations.

Select tuples from a relation satisfying a predicate

- Results in a new relation that is a subset of the original.
- Why is it useful that the result Is a relation?

Relational Algebra - Basic Operations

Basic operators

- select: σ
- Project: **Π**
- Natural join: ⋈
- union: ∪
- set difference: -
- Cartesian product: ×
- Rename: ρ

The Select Operation

- Notation is $\sigma_p(x)$
  - P is the selection predicate
  - x is the relation

- p is a boolean formula of terms and connectives.

- Connectives: and ($\wedge$), or ($\vee$), and not ($\neg$).
- Operators: $<$, $>$, $\leq$, $\geq$, $=$, $\neq$
- Terms:
  - attribute operator attribute
  - attribute operator constant

The Select Operation

Notation is $\sigma_p(x)$

$\sigma_{salary >= 85000}$ (instructor)

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

The *instructor* relation

Select the tuples with attribute salary at least 85000 from the instructor relation

The Select Operation - resulting relation

Notation is $\sigma_p(x)$

$\sigma_{salary >= 85000}$ (instructor)

ID	*name*	*dept_name*	*salary*
12121	Wu	Finance	90000
22222	Einstein	Physics	95000
33456	Gold	Physics	87000
83821	Brandt	Comp. Sci.	92000

Result of query selecting instructor tuples with salary greater than $85000

Select the tuples with attribute salary at least 85000 from the instructor relation

The Project Operation

Symbol is Π

Selection of attributes

$\Pi_{\text{ID, name, salary}}$ (instructor)

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

The *instructor* relation

Select all the tuples from the instructor relation with attributes ID, name and salary

The Project Operation - resulting relation

Symbol is Π

Selection of attributes

$\Pi_{\text{ID, name, salary}}$ (instructor)

ID	*name*	*salary*
10101	Srinivasan	65000
12121	Wu	90000
15151	Mozart	40000
22222	Einstein	95000
32343	El Said	60000
33456	Gold	87000
45565	Katz	75000
58583	Califieri	62000
76543	Singh	80000
76766	Crick	72000
83821	Brandt	92000
98345	Kim	80000

Result of $\Pi_{\text{ID, name, salary}}$ (instructor).

Select all the tuples from the instructor relation with attributes ID, name and salary

Composition of Relational Operations

- The result of a relational-algebra operation is relation
- Result of can be composed together into a relational-algebra expression.
- Consider the query –
  - Find the names of all instructors in the Physics department.

$$\Pi_{\text{name}} (\sigma_{\text{dept name} = \text{“Physics”}} (\text{instructor}))$$

- Just like composing arithmetic operations (such as +, −, ∗, and ÷) into arithmetic expressions.

The Cartesian Product Operation

- The Cartesian-product operation (denoted by ×) allows us to combine information from any two relations.
- This is the cross product of two relations.

- **What is the cross product of {a, b} and {c, d}?**
  - {a, b} X {c, d} produces {(a, c), (a, d), (b, c), (b, d)}

The cross product produces all possible pairs of rows of the two relations.

- **Can you see a problem?**
  - If the two relations have attributes in common, how do we tell which relation each attribute is from?

The Cartesian Product Operation - Without Common attribute

Relations: r,s

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

r X s

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

The Cartesian Product Operation - Common attribute

Relations: r,s

A	B
α	1
β	2

r

B	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

r X s

A	r.B	s.B	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Solution:

Renaming Attributes

Allows us to refer to a relation, by more than one name.

Example

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

The *instructor* relation

ID	course_id	sec_id	semester	year
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

The *teaches* relation

The Cartesian Product Operation - Issue

Example: the Cartesian product of the relations *instructor* and *teaches* is written as:

instructor × *teaches*

We construct a tuple of the result out of each possible pair of tuples:

one from the *instructor* relation and one from the *teaches* relation (see next slide)

The instructor *ID* appears in both relations

How do we distinguish these attributes?

Example

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

The *instructor* relation

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

The *teaches* relation

The Cartesian Product Operation - Renaming Attributes

The instructor *ID* appears in both relations we distinguish between these attribute by attaching to the attribute the name of the relation from which the attribute originally came.

- *instructor.ID*
- *teaches.ID*

the relation schema for instructor × teaches is:

(*instructor*.ID, *instructor*.name, *instructor*.dept name, *instructor*.salary, teaches.ID, teaches.course id, teaches.sec id, teaches.semester, teaches.year)

Result of the Cartesian product *instructor × teaches*

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	12121	FIN-201	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	15151	MU-199	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
12121	Wu	Finance	90000	10101	CS-101	1	Fall	2017
12121	Wu	Finance	90000	10101	CS-315	1	Spring	2018
12121	Wu	Finance	90000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
12121	Wu	Finance	90000	15151	MU-199	1	Spring	2018
12121	Wu	Finance	90000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
15151	Mozart	Music	40000	10101	CS-101	1	Fall	2017
15151	Mozart	Music	40000	10101	CS-315	1	Spring	2018
15151	Mozart	Music	40000	10101	CS-347	1	Fall	2017
15151	Mozart	Music	40000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
15151	Mozart	Music	40000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
22222	Einstein	Physics	95000	10101	CS-101	1	Fall	2017
22222	Einstein	Physics	95000	10101	CS-315	1	Spring	2018
22222	Einstein	Physics	95000	10101	CS-347	1	Fall	2017
22222	Einstein	Physics	95000	12121	FIN-201	1	Spring	2018
22222	Einstein	Physics	95000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...

The Join Operation - Need

- The Cartesian-Product

  instructor × teaches

  associates every tuple of instructor with every tuple of teaches.
- Most of the resulting rows have information about instructors who did NOT teach a particular course.
- To get only those tuples of "instructor × teaches " that pertain to instructors and the courses that they taught, we write:

  $\sigma_{instructor.ID = teaches.ID}(instructor \times teaches)$.

- We get only those tuples of "instructor X teaches" that pertain to instructors and the courses that they taught.

RESULT $\sigma_{instructor.ID = teaches.ID}(instructor \times teaches)$.

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-319	2	Spring	2018
98345	Kim	Elec. Eng.	80000	98345	EE-181	1	Spring	2017

Join Operation

Two types: Natural Join and Theta Join

- Natural Join : $\bowtie$
    - Binary operator
    - Combine two relations into a single relation.
- Theta Join : $\bowtie_{\theta}$
    - Binary operator
    - Combine two relations into a single relation.
    - θ acts as a predicate/condition
    - The tuples are joined if the predicate is satisfied

Natural Join

- The tuples are joined if the attributes common to both relations are equal
- Example:
- Two relations R(a, b, c) and S (a, d)
  - Equivalence: R ⋈ S is equivalent to $\sigma_{R.a = S.a}(R \times S)$.

R

a	b	c
1	2	3
6	5	4
9	8	7

S

a	d
1	2
4	3
6	5

Theta Join

- The tuples are joined if the predicate is satisfied
- Example:
- Two relations R(a, b, c) and S (a, d)
  - Equivalence: $R \bowtie_{R.b = S.d} S$ is equivalent to $\sigma_{R.b = S.d}(R \times S)$.

R

a	b	c
1	2	3
6	5	4
9	8	7

S

a	d
1	2
4	3
6	5

Note:
Predicate can be any condition/any comparison.
It is not limited to checking equality

The Natural Join Example

- The Natural join operation allows us to combine a select operation and a Cartesian-Product operation into a single operation.

$\sigma_{instructor.ID = teaches.ID}(instructor \times teaches)$ → $instructor \bowtie_{instructor.ID=teaches.ID} teaches$

The Union Operation

Symbol: ∪

Relations r, s:

For r ∪ s to be valid:

1. r, s must have the same arity
   (same number of attributes)
2. The attribute domains must be compatible
   i.e, 2nd column of r deals with the same type of values as does the 2nd column of s.

Q. Did you expect there to be 4 rows?

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

r ∪ s

A	B
α	1
α	2
β	1
β	3

section relation

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

The Union Operation Query

Query: find the set of all courses taught in the Fall 2009 semester, the Spring 2010 semester, or both.

To find the set of all courses taught in the Fall 2009 and Spring 2010 semester:

$\Pi_{course\_id} (\sigma_{semester = \text{"Fall"} \wedge year=2009} (section))$

$\Pi_{course\_id} (\sigma_{semester = \text{"Spring"} \wedge year=2010} (section))$

Query:

$\Pi_{course\_id} (\sigma_{semester = \text{"Fall"} \wedge year=2009} (section)) \cup \Pi_{course\_id} (\sigma_{semester = \text{"Spring"} \wedge year=2010} (section))$

$\Pi_{course\_id}(\sigma_{semester = \text{"Fall"} \land year=2009}(section)) \cup \Pi_{course\_id}(\sigma_{semester = \text{"Spring"} \land year=2010}(section))$

The Union Operation Query result

course_id
CS-101
CS-315
CS-319
CS-347
FIN-201
HIS-351
MU-199
PHY-101

Result of Courses offered in either Fall 2009, Spring 2010, or both semesters.

The Intersection Operation

What would you expect them to be?

Relations r, s:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

$r \cap s$

A	B
α	2

The Intersection Operation Query

- The set-intersection operation allows us to find tuples that are in both the input relations.
- Notation: $r \cap s$
- Assume:
  - r, s have the same arity
  - attributes of r and s are compatible

Query: Find the set of all courses taught in both the Fall 2009 and the Spring 2010 semesters.

$$\Pi_{course\_id} (\sigma_{semester = \text{"Fall"} \land\ year=2009} (section)) \cap \Pi_{course\_id} (\sigma_{semester = \text{"Spring"} \land\ year=2010} (section))$$

$\Pi_{course\_id}(\sigma_{semester=\text{"Fall"} \land year=2009}(section)) \cap \Pi_{course\_id}(\sigma_{semester=\text{"Spring"} \land year=2010}(section))$

The Intersection Operation Query result

course_id
CS-101

Courses offered in both the Fall 2009 and Spring 2010 semesters.

The Difference Operation

What would you expect them to be?
Relations r, s:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

r – s

A	B
α	1
β	1

The Difference Operation - Query

- The set-difference operation allows us to find tuples that are in one relation but are not in another.
- Notation r – s
- Set differences must be taken between compatible relations.
  - r and s must have the same arity
  - attribute domains of r and s must be compatible
- Query: to find all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

$$\Pi_{course\_id} (\sigma_{semester = \text{“Fall”} \wedge \ year=2009} (section)) - \Pi_{course\_id} (\sigma_{semester = \text{“Spring”} \wedge \ year=2010} (section))$$

$\Pi_{course\_id}(\sigma_{semester = \text{“Fall”} \wedge year=2009}(section)) - \Pi_{course\_id}(\sigma_{semester = \text{“Spring”} \wedge year=2010}(section))$

The Difference Operation - Query result

course_id
CS-347
PHY-101

Courses offered in the Fall 2009 semester but not in Spring 2010 semester.

The Rename Operation

- Symbol ϱ
- The results of relational-algebra expressions do not have a name that we can use to refer to them.
- The rename operator, ϱ , is provided for that purpose
- The expression:

$$\varrho_X (E)$$

returns the result of expression *E* under the name *x*

Need of rename (ρ)

Query:

Find the *ID* and *name* of those instructors who earn more than the instructor whose ID is 12121.

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

Unary operators: rename (ρ)

Query:

Find the ID and name of those instructors who earn more than the instructor whose ID is 12121.

Difficulty:

Reference the instructor relation once to get the salary of instructor 12121 (w) and then a second time to get the salary of each instructor (i)

Step 1: Rename the Table: ρ_w (instructor)

Step 2: Select tuple for instructor with ID 12121

$$\sigma_{w.id=12121}(\rho_w \text{ (instructor)})$$

Step 3: Rename the Table: ρ_i (instructor)

Step 4: Get the cartesian product for comparison

$$\rho_i \text{ (instructor)} \times \sigma_{w.id=12121}(\rho_w \text{ (instructor)})$$

Unary operators: rename (ρ)

Query:

Find the ID and name of those instructors who earn more than the instructor whose ID is 12121.

Difficulty:

Reference the instructor relation once to get the salary of each instructor (i) and then a second time to get the salary of instructor 12121 (w)

Step 5: Select the tuples such that salary of instructors is greater than salary of instructor with ID 12121

$$\sigma_{i.salary > w.salary}(\rho_i(\text{instructor}) \times \sigma_{w.id=12121}(\rho_w(\text{instructor}))$$

Final step: Project only ID and name of those instructors

The Rename Operation Query

- Query:
  - Find the ID and name of those instructors who earn more than the instructor whose ID is 12121.

Query:

$$\Pi_{i.ID,i.name}((\sigma_{i.salary > w.salary}(\rho_i(instructor) \times \sigma_{w.id=12121}(\rho_w\ (instructor)))))$$

Next week

Introduction to SQL..