فصل دوم: آشنایی با مدل رابطه ای

درس پایگاه داده
دانشگاه صنعتی نوشیروانی بابل
مهدی عمادی
m.emadi@nit.ac.ir

فصل دوم: آشنایی با مدل رابطه ای

- ساختار پایگاه داده رابطه ای (Relational Databases)
- شمای پایگاه داده (Database Schema)
- کلیدها (Keys)
- نمودارهای شما (Schema Diagrams)
- زبان پرس و جوی رابطه ای (Relational Query Languages)
- جبر رابطه ای (The Relational Algebra)

مثال رابطه *Instructor*

خصیصه یا فیلد (ستون)

attributes
(or columns)

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

تاپل (سطر یا رکورد)

tuples
(or rows)

صفت (Attribute)

- ستون (خصیصه یا فیلد)
- دامنه (Domain) :
  - مجموعه مقادیر مجاز برای یک صفت
- صفت های اتمیک (Atomic)
  - یک صفت اتمیک است اگر مقادیر مجاز آن صفت قابل تجزیه به چند بخش با معنی (یا صفت) نباشد
- هیچ مقدار (null value)
  - هیچ مقدار یک مقدار قابل قبول برای همه دامنه ها می باشد.
  - در حقیقت بیان کننده مشخص نبودن (unknown) مقدار یک صفت است.
  - هیچ مقدار مسائل زیادی را در تعریف عملگرها ایجاد می کند.

رابطه ها ترتیب ندارند

- تاپل های رابطه ممکن است به هر ترتیبی ذخیره شوند و لزوما مرتب شده بر اساس هیچ یک از صفات نیستند.
- مثال: رابطه ***instructor***

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

شمای پایگاه داده (Database Schema)

- به ساختار منطقی پایگاه داده شمای پایگاه داده گفته می شود.
- نمونه پایگاه داده (Database instance) : یک تصویر لحظه ای (snapshot) از داده های موجود در پایگاه داده که در یک زمان خاص از وضعیت پایگاه داده گرفته می شود.
- مثال:

- **schema:** *instructor* (*ID, name, dept_name, salary*)
- **Instance:**

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

انواع کلیدها

- ابر کلید SK (Super Key)
- کلید کاندید CK (Candidate Key)
- کلید اصلی PK (Primary Key)
- کلید دیگر AK (Alternate Key)
- کلید خارجی FK (Foreign Key)

ابر کلید SK (Super Key)

- هر زیر مجموعه از مجموعه عنوان رابطه که یکتایی مقدار در گسترده رابطه داشته باشد. به بیان دیگر، هر ترکیبی از اسامی صفات رابطه که در هیچ دو سطر جدول معادل آن مقدار یکسان نداشته باشد.
- کد ملی
- در نرمال سازی رابطه ها کاربرد دارد.

کلید کاندید (Candidate Key) CK

- یکتایی مقدار دارد.
- کاهش ناپذیر باشد، یعنی اگر یکی از عناصر آن زیرمجموعه را حذف کنیم، زیرمجموعه باقیمانده دیگر خاصیت یکتایی مقدار نداشته باشد.
- هر ابر کلید لزوما کلید کاندید نیست، اما هر کلید کاندید، جزء مجموعه ابر کلیدهای رابطه هست.
- کاربرد اصلی آن در هنگام Query است.

کلید اصلی PK (Primary Key)

- یکی از کلیدهای کاندید رابطه که طراح انتخاب می کند و به سیستم معرفی می کند.
- چون هر رابطه ای حداقل یک کلید کاندید دارد، بنابراین هر رابطه ای حتما یک کلید اصلی دارد.
- در بعضی مواقع طراح خود یک صفت جدید را که قالبا ساده تر از کلیدهای کاندید است را به رابطه اضافه می کند و آن را به عنوان کلید اصلی معرفی می کند.

کلید دیگر (Alternate Key) AK

- هر کلید کاندید، غیر از کلید اصلی، کلید دیگر نام دارد.
- اگر همه کلیدهای کاندید رابطه و نیز خود کلید اصلی به سیستم معرفی شوند، دیگر نیاز به تصریح کلید دیگر نیست.
- اما در بیشتر اوقات یک کلید اصلی معرفی می شود و فقط چند کلید کاندید مورد نظر به شکل کلید دیگر معرفی می شوند.

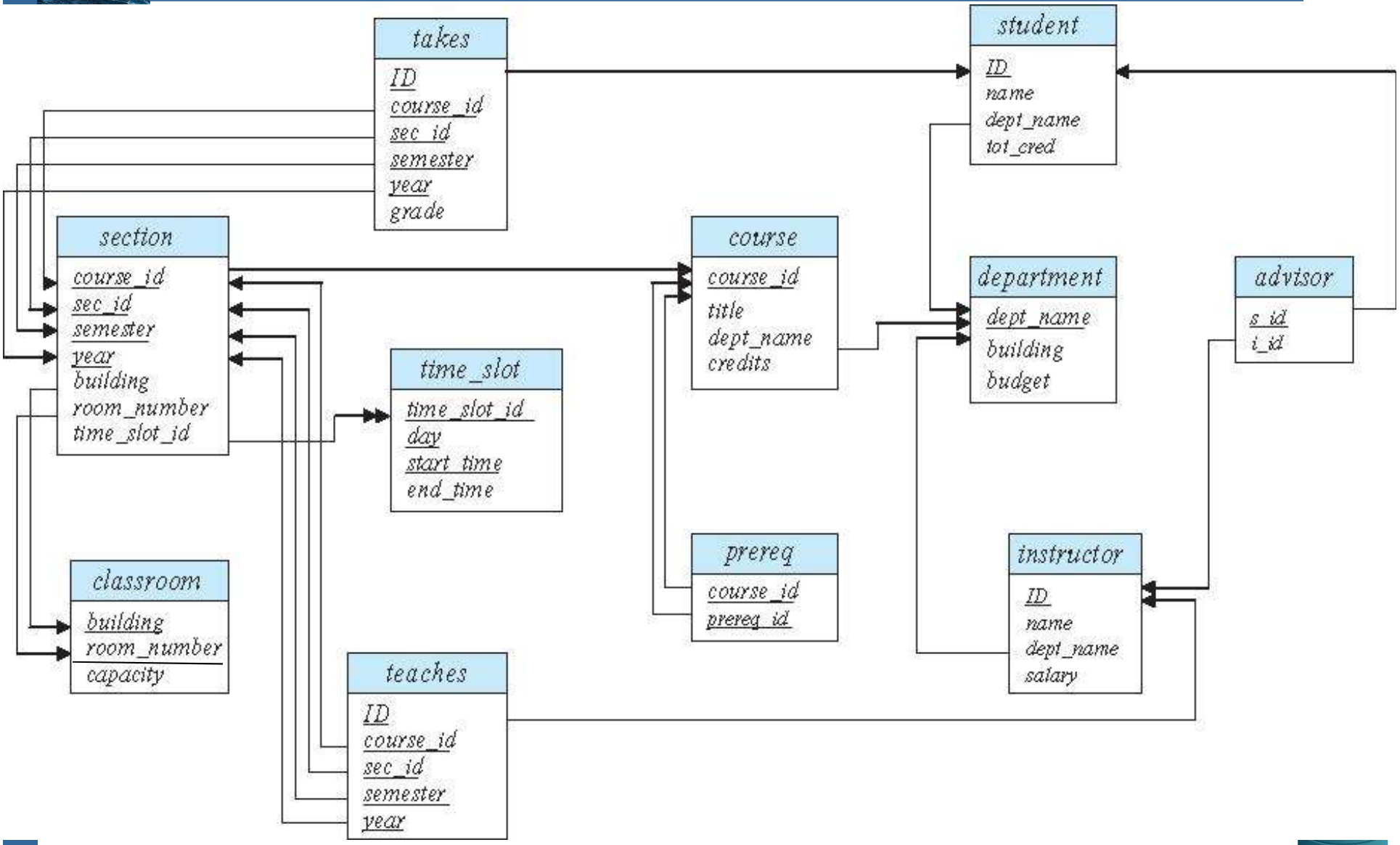
کلید خارجی FK (Foreign Key)

- دو رابطه **R1** و **R2** را در نظر می کیریم. هر زیرمجموعه از صفات رابطه **R2** که در رابطه **R1** کلید کاندید باشد، کلید خارجی در رابطه **R2** است.
- نقش کلید خارجی چیست؟
  - کلید خارجی برای نمایش ارتباطات بین انواع جدولها یا رابطه ها بکار می رود.
- بحث کامل تر کلید خارجی و ارجاع (**Reference**) را به همراه قواعد جامعیتی بیان خواهیم کرد.

مثال پایگاه داده دانشگاه

- در این درس ما از پایگاه داده دانشگاه برای نشان دادن تمام مفاهیم استفاده خواهیم کرد
- شامل:
  - دانشجویان (Students)
  - مدرسان (Instructors)
  - کلاس ها (Classes)
- مثال برنامه مورد نیاز:
  - افزودن دانشجویان، مدرسان و درس های جدید
  - ثبت نام دانش آموزان برای درس ها، و ایجاد لیست دانشجویان کلاس
  - ثبت نمرات دانشجویان
  - محاسبه معدل (GPA) و ساختن کارنامه دانشجو

Schema Diagram for University Database



زبان پرس و جوی رابطه ای (Relational Query Languages)

- رویه ای (Procedural) در مقابل نارویه ای (non-procedural) یا توصیفی (declarative)
  - جبر رابطه ای (Relational algebra) – رویه ای
  - حساب رابطه ای تاپل (Tuple relational calculus) – نارویه ای
  - حساب رابطه ای دامنه (Domain relational calculus) – نارویه ای
- قدرت محاسباتی هر سه زبان فوق برابر است
- یک پرس وجو بر یک نمونه از رابطه اعمال می‌شود و حاصل آن نیز یک رابطه است
  - شمای رابطه ورودی به یک پرس و جو ثابت می باشد.
  - شمای رابطه حاصل از یک پرس و جو نیز ثابت است.
- در این فصل تمرکز بر جبر رابطه ای است
  - معادل تورینگ ماشین نیست.
  - شامل ۶ عملگر پایه ای است.

جبر رابطه ای (Relational Algebra)

- یک زبان رویه ای است. شامل تعدادی عملگر می شود که هر عملگر یک یا دو رابطه را به عنوان ورودی دریافت و یک رابطه جدید به عنوان خروجی آن تولید می کند.
- شش عملگر پایه
  - گزینش، تحدید یا ***Selection*** (σ): گزینش یک زیرمجموعه از تاپل‌ها (سطرهای) یک رابطه (جدول)
  - پرتو یا ***Projection*** (Π): نمایش ستون‌های خاص از یک رابطه (جدول)
  - ضرب کارتزین یا ***Cross-product*** یا ***Cartesian-product*** (X): امکان ترکیب دو رابطه (جدول)
  - تفاضل یا ***Set-difference*** ($-$): نمایش ستون‌های خاص از یک رابطه (جدول)
  - اجتماع یا ***Union*** ($\cup$): نمایش ستون‌های خاص از یک رابطه (جدول)
  - تغییر نام یا دگر نامی یا ***renaming*** (ρ)
- دیگر عملگرها
  - پیوند یا Join ($\blacktriangleleft\blacktriangleright$)، تقسیم یا division ($\div$)، اشتراک یا Intersection ($\cap$)

عملگر گزینش، تحدید یا انتخاب (Select Operation)

- انتخاب سطرهایی که شرط گزینش (*selection condition* یا *Selection predicate*) را برآورده کند

- **Notation:** $\sigma_p(r)$
- ***p* is called the selection predicate**

- مثال: سطرهایی از رابطه *instructor* را انتخاب نمایید که مدرس عضو گروه فیزیک (Physics) باشد

- **Example: select those tuples of the *instructor* relation where the instructor is in the "Physics" department.**

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

Query

$$\sigma_{dept_name=\text{“}Physics\text{”}}(instructor)$$

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Select Operation (Cont.)

- We allow comparisons using

  $=, \neq, >, \geq, <, \leq$

  in the selection predicate.
- We can combine several predicates into a larger predicate by using the connectives:

  $\wedge$ (and), $\vee$ (or), $\neg$ (not)
- Example: Find the instructors in Physics with a salary greater \$90,000, we write:

$$\sigma_{dept_name = \text{“Physics”} \wedge salary > 90{,}000}(instructor)$$

- Then select predicate may include comparisons between two attributes.
  - Example, find all departments whose name is the same as their building name:
  - $\sigma_{dept_name = building}(department)$

عملگر پرتو (Project Operation)

- پرتو یک عملگر یگانی (unary) است. ستون هایی که در لیست پرتو نباشند در خروجی حذف می‌شوند
- عملگر پرتو باید تکراری‌ها (duplicates) را حذف کند. چرا؟
- نکته: در عمل در سیستم‌های کاربرد موجود تکراری‌ها حذف نمی‌شوند مگر کاربر به طور مستقیم اعلام کند

- **Notation:**

$$\Pi_{A_1,A_2,A_3 \ldots A_k}(r)$$

where A_1, A_2 are attribute names and r is a relation name.

Project Operation (Cont.)

- **Example: eliminate the *dept_name* attribute of *instructor***
- **Query:**

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

- **Result:**

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

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

یک نمونه پایگاه داده رابطه ای

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

ترکیب عملگرهای رابطه ای

- خروجی عملیات جبر رابطه ای خود یک رابطه است بنابراین می توان عملیات جبر رابطه ای دیگری بر روی آن اعمال نمود. به طور کلی هر ترکیبی از عملگرها را می توان به صورت یک عبارت جبر رابطه ای (relational-algebra expression) بیان نمود.
- مثال: نام تمام اساتید دانشکده فیزیک را بیابید:

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

- به جای نام یک رابطه می توان یک عبارت جبر رابطه ای را به عنوان ورودی به یک عملگر داد.

Cartesian-Product Operation

- یک عملگر دودویی (binary) است که هر سطر از یک رابطه با هر سطر از رابطه دیگر به صورت زوج‌هایی با یکدیگر سطری جدید ایجاد می‌کنند.
- در شمای نتیجه به ازای هر فیلد در رابطه ها یک فیلد داریم.
- تداخل: اگر دو رابطه فیلدی با نام مشترک داشته باشند!!
- با X نمایش داده می شود.

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

  instructor* X *teaches

- تداخل: اگر دو رابطه فیلدی با نام مشترک داشته باشند!!
  - ***instructor.ID***
  - ***teaches.ID***

The *instructor* x *teaches* table

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
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…

عملگر پیوند (Join Operation)

- The Cartesian-Product

  $$instructor \times teaches$$

  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* X *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.
- The result of this expression, shown in the next slide

Join Operation (Cont.)

- The table corresponding to:

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

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 (Cont.)

- The **join** operation allows us to combine a select operation and a Cartesian-Product operation into a single operation.
- Consider relations r (R) and s (S)
- Let "theta" be a predicate on attributes in the schema R "union" S. The join operation r $\bowtie_\theta$ s is defined as follows:

$$r \bowtie_\theta s = \sigma_\theta (r \times s)$$

- Thus

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

- Can equivalently be written as

$$instructor \bowtie_{Instructor.id = teaches.id} teaches.$$

عملگر اجتماع (Union Operation)

- The union operation allows us to combine two relations
- Notation: $r \cup s$
- For $r \cup s$ to be valid.

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

- Example: to find all courses taught in the Fall 2017 semester, or in the Spring 2018 semester, or in both

$$\Pi_{course_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) \cup \Pi_{course_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$$

Union Operation (Cont.)

- **Result of:**

$$\Pi_{course_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) \cup \Pi_{course_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$$

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

(Set-Intersection Operation) عملگر اشتراک

- 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
- Example: Find the set of all courses taught in both the Fall 2017 and the Spring 2018 semesters.

$$\Pi_{course_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) \cap \Pi_{course_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$$

- Result

course_id
CS-101

(Set Difference Operation) عملگر تفاضل

- 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
- Example: to find all courses taught in the Fall 2017 semester, but not in the Spring 2018 semester

$$\Pi_{course_id}(\sigma_{semester=\text{“Fall”} \wedge year=2017}(section)) - \Pi_{course_id}(\sigma_{semester=\text{“Spring”} \wedge year=2018}(section))$$

course_id
CS-347
PHY-101

The Assignment Operation

- It is convenient at times to write a relational-algebra expression by assigning parts of it to temporary relation variables.
- The assignment operation is denoted by ← and works like assignment in a programming language.
- Example: Find all instructor in the "Physics" and Music department.

$$Physics \leftarrow \sigma_{dept_name=\text{“}Physics\text{”}}(instructor)$$

$$Music \leftarrow \sigma_{dept_name=\text{“}Music\text{”}}(instructor)$$

$$Physics \cup Music$$

- With the assignment operation, a query can be written as a sequential program consisting of a series of assignments followed by an expression whose value is displayed as the result of the query.

The Rename Operation

- 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:

$$\rho_x(E)$$

returns the result of expression E under the name x

- Another form of the rename operation:

$$\rho_{x(A1,A2,..An)}(E)$$

Equivalent Queries

- There is more than one way to write a query in relational algebra.
- Example: Find information about courses taught by instructors in the Physics department with salary greater than 90,000
- Query 1

$$\sigma_{dept_name=\text{“Physics”} \wedge salary > 90{,}000}(instructor)$$

- Query 2

$$\sigma_{dept_name=\text{“Physics”}}(\sigma_{salary > 90.000}(instructor))$$

- The two queries are not identical; they are, however, equivalent -- they give the same result on any database.

Equivalent Queries

- There is more than one way to write a query in relational algebra.
- Example: Find information about courses taught by instructors in the Physics department
- Query 1

$$\sigma_{dept_name=\text{“Physics”}}(instructor \bowtie_{instructor.ID = teaches.ID} teaches)$$

- Query 2

$$(\sigma_{dept_name=\text{“Physics”}}(instructor)) \bowtie_{instructor.ID = teaches.ID} teaches$$

- The two queries are not identical; they are, however, equivalent -- they give the same result on any database.

پایان فصل دوم

Examples of Division A/B

sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

A

pno
p2

B1

sno
s1
s2
s3
s4

A/B1

pno
p2
p4

B2

sno
s1
s4

A/B2

pno
p1
p2
p4

B3

sno
s1

A/B3

Select Operation – Example

- Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Project Operation – Example

Relation *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

$\Pi_{A,C}\ (r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Union Operation – Example

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
α	2
β	1
β	3

Set Difference Operation – Example

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

Cartesian-Product Operation – Example

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

Composition of Operations

- Can build expressions using multiple operations
- Example: $\sigma_{A=C}(r \times s)$
- $r \times s$

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

- $\sigma_{A=C}(r \times s)$

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

Set-Intersection Operation – Example

Relation *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

Natural Join Operation – Example

- **Relations r, s:**

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	$\in$

s

- r ⋈ s

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

Division Operation – Example

- Relations *r, s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
ϵ	6
ϵ	1
β	2

r

B
1
2

s

- $r \div s$:

A
α
β

Another Division Example

- Relations *r, s*:

A	B	C	D	E
α	a	α	a	1
α	a	γ	a	1
α	a	γ	b	1
β	a	γ	a	1
β	a	γ	b	3
γ	a	γ	a	1
γ	a	γ	b	1
γ	a	β	b	1

r

D	E
a	1
b	1

s

- $r \div s$:

A	B	C
α	a	γ
γ	a	γ

Outer Join – Example

- **Relation *loan***

loan_number	*branch_name*	*amount*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

- Relation *borrower*

customer_name	*loan_number*
Jones	L-170
Smith	L-230
Hayes	L-155

Outer Join – Example

- **Inner Join**

  $loan \bowtie Borrower$

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

- Left Outer Join

  $loan \mathbin{⟕} Borrower$

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*

Outer Join – Example

- Right Outer Join

  loan ⟖ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	*null*	*null*	Hayes

- Full Outer Join

  loan ⟗ *borrower*

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*
L-155	*null*	*null*	Hayes