关系模型
Relational Model

李文根/Wengen Li

Email: lwengen@tongji.edu.cn

先进数据与机器智能系统实验室（ADMIS）
https://admis.tongji.edu.cn/main.htm

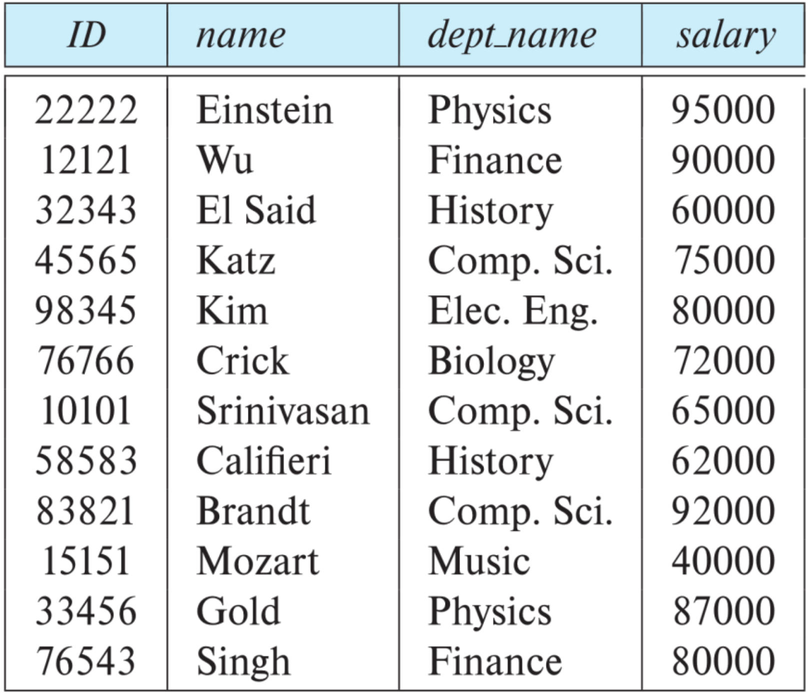
课程概要

- **Part 0: Overview**
  - Ch1: Introduction
- **Part 1 Relational Databases**
  - **Ch2: Relational model (data model, relational algebra)**
  - Ch3&4: SQL(Structured Query Language)
  - Ch5: Advanced SQL
- **Part 2 Database Design**
  - Ch6: Database design based on E-R model
  - Ch7: Relational database design
- **Part 3 Application Design & Development**
  - Ch8: Complex data types
  - Ch9: Application development
- **Part 4 Big data analytics**
  - Ch10: Big data
  - Ch11: Data analytics
- **Part 5 Data Storage & Indexing**
  - Ch12: Physical storage system
  - Ch13: Data storage structure
  - Ch14: Indexing
- **Part 6 Query Processing & Optimization**
  - Ch15: Query processing
  - Ch16: Query optimization
- **Part 7 Transaction Management**
  - Ch17: Transactions
  - Ch18: Concurrency control
  - Ch19: Recovery system
- **Part 8 Parallel & Distributed Database**
  - Ch20: Database system architecture
  - Ch21-23: Parallel & distributed storage, query processing & transaction processing
- **Part 9**
  - DB Platform: **OceanBase**, MongoDB, Neo4J

目录

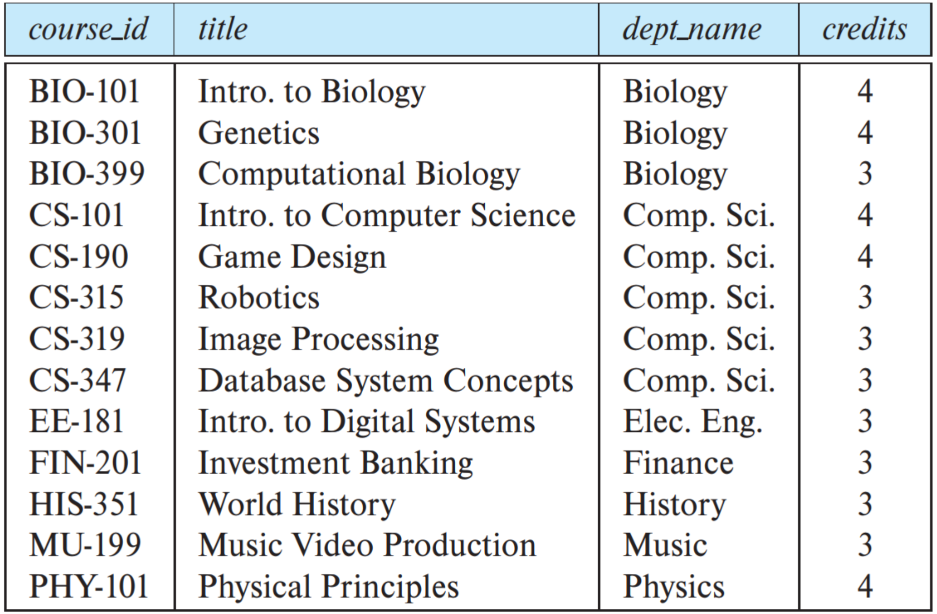
- **关系数据库的结构**
- 数据库模式
- 码
- 模式图
- 关系查询语言
- 关系代数

关系表示例



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

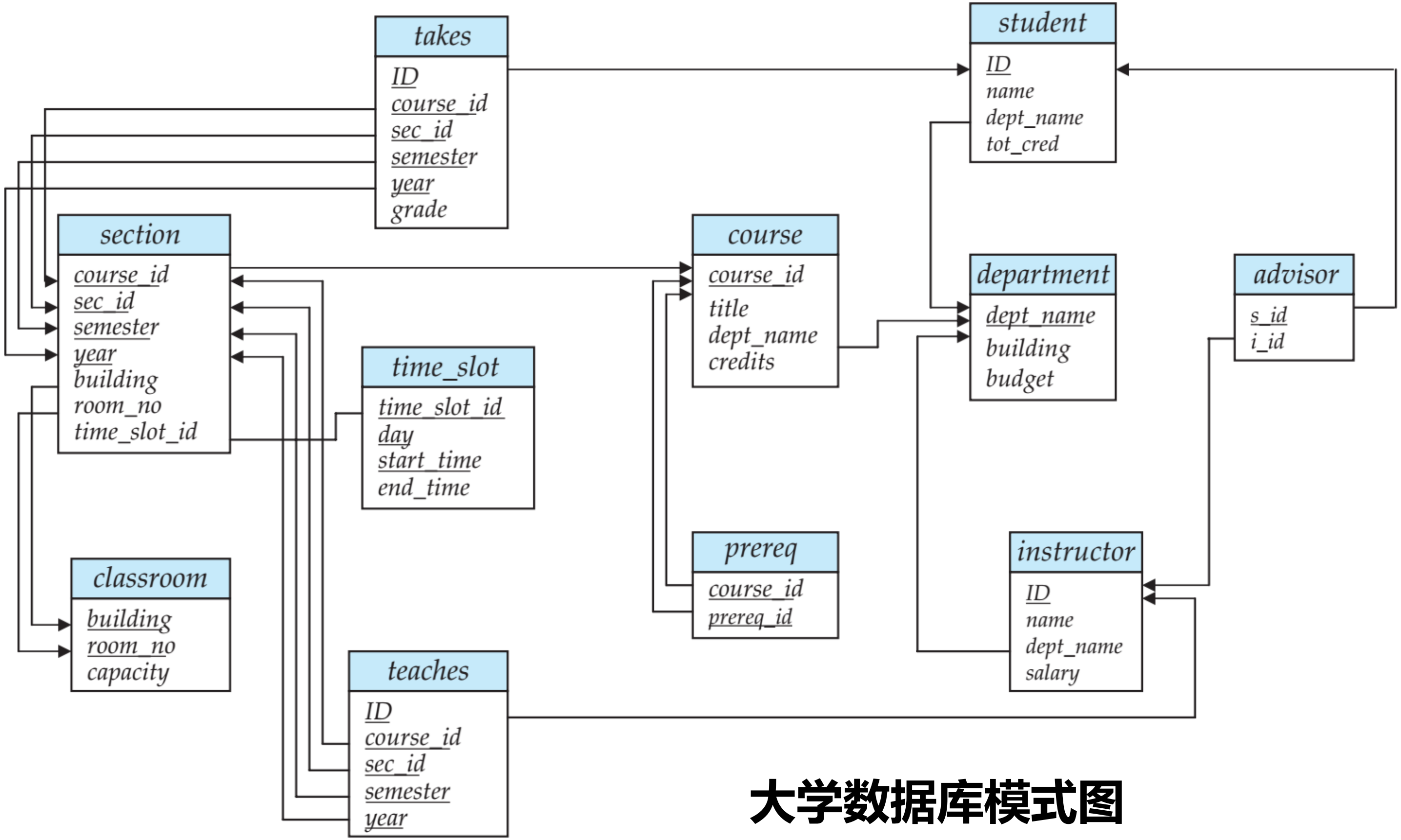
Instructor



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

Course

关系数据库示例



大学数据库模式图

关系(Relation)的基本结构

TONGJI UNIVERSITY

- Given sets $D_1, D_2, \cdots, D_n$, a relation r is a subset of $D_1 \times D_2 \times \cdots \times D_n$, i.e., a set of n-tuples $(a_1, a_2, \cdots, a_n)$ where each $a_i \in D_i(i = 1, \ldots n)$

- 例如:

customer_name = {Jones, Smith, Curry, Lindsay}
customer_street = {Main, North, Park}
customer_city   = {Harrison, Rye, Pittsfield}

则:

r = {(Jones, Main, Harrison), (Smith, North, Rye), (Curry, North, Rye),
(Lindsay, Park, Pittsfield)}

is a relation over **customer_name** x **customer_street** x **customer_city**

属性(Attribute)

- 每个关系由一组属性组成：$A_1, A_2, \dots, A_n$
- **属性的域(Domain)**
  - the whole set of available and legal values of the attribute
- **属性的值**
  - 通常要求是原子的(**atomic**)
  - 多值属性和复合属性不是原子的
  - **空值(Null value):** 表示值未知或不存在

关系模式(Relation Schema)

- $A_1, A_2, \dots, A_n$are attributes, and $R = (A_1, A_2, \dots, A_n)$ is a relation schema
- 例如:

instructor-schema =(id, name, dept_name, salary)

- $r(R)$ is a relation on the relation schema R
- 例如:

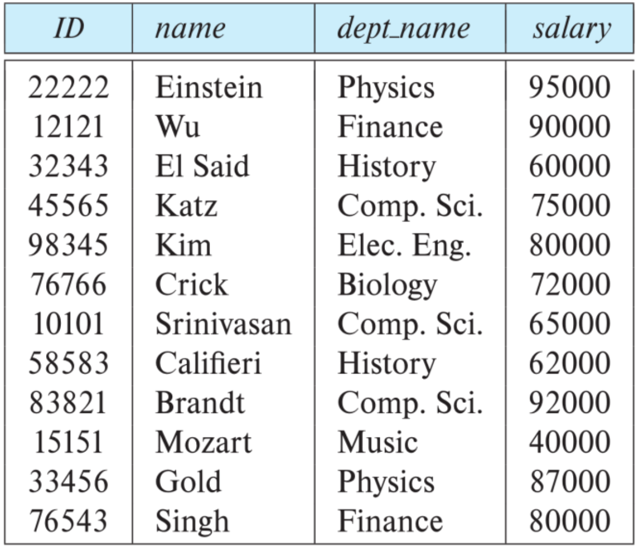
customer(customer_schema)

关系实例(Relation Instance)

- The current values (i.e., relation instance) of a relation are specified by a table
- An element t of relation r is a tuple (元组), represented by a row in the table
- The order of tuples/attributes in a relation is irrelevant (无关紧要的). Tuples could be stored in an arbitrary order

Attributes/Columns

Tuples/Rows



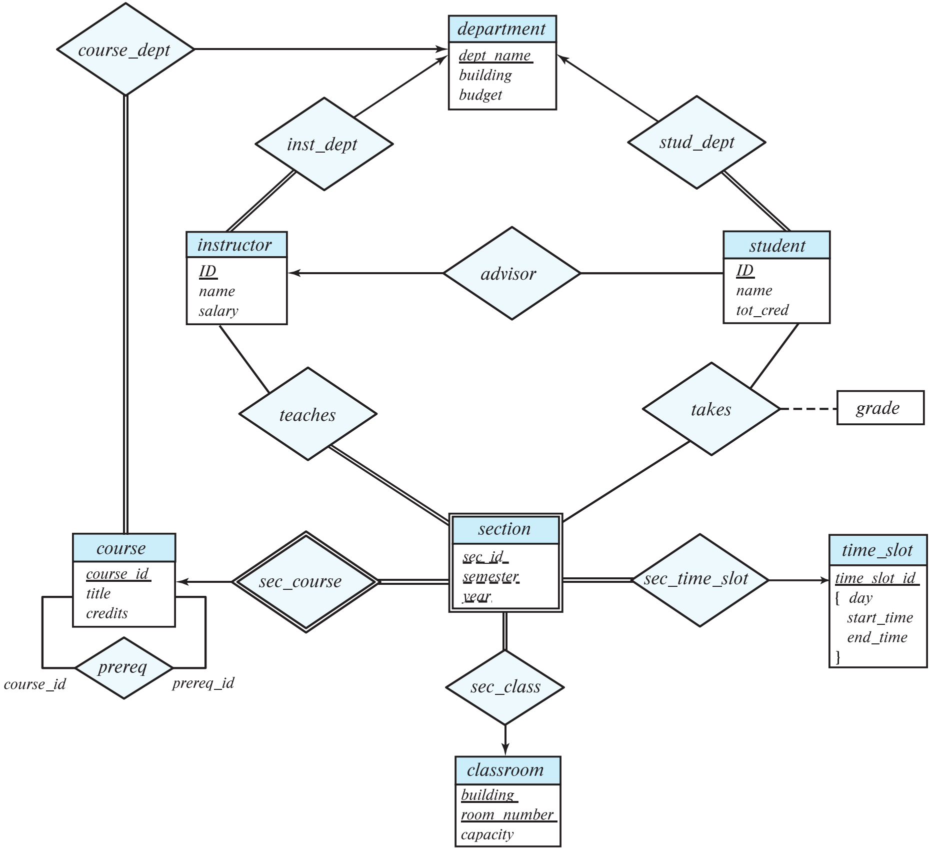
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

目录

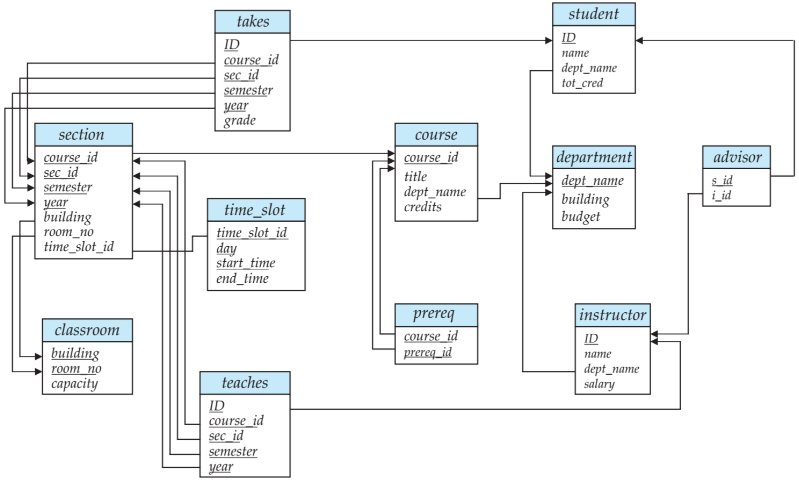
- 关系数据库的结构
- **数据库模式**
- 码
- 模式图
- 关系查询语言
- 关系代数

数据库模式(Database Schema)

- A database consists of multiple relations
- Storing all information as a single relation results in
  - **数据冗余**：repetition of information, e.g., one department has many students, record the information of both department and student
  - **数据稀疏**：the need for null values, e.g., represent a student with only a few courses
- Normalization(规范化) theory (Chapter 7) discusses the ways to design relational schemas



E-R Diagram for University Database



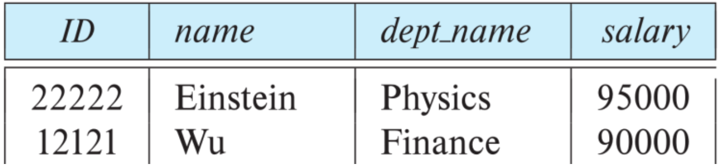
Schema for University Database

目录

- 关系数据库的结构
- 数据库模式
- **码**
- 模式图
- 关系查询语言
- 关系代数

Keys (码、键)

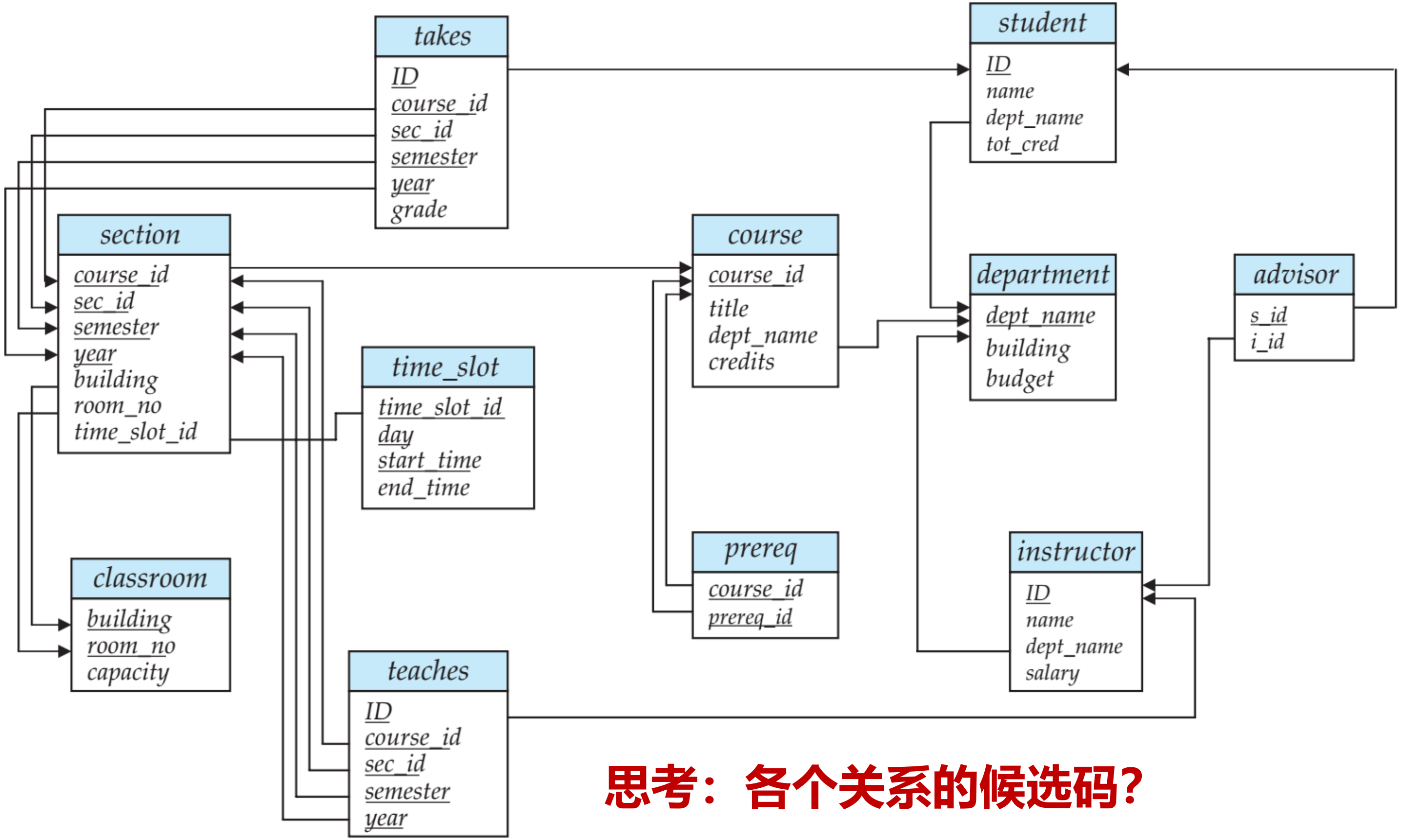
- **超码(Superkey)**
  - Let $K \subseteq R$, K is a superkey of relation schema R if the values of K are sufficient to identify a unique tuple of each possible relation $r(R)$
  - E.g.,{*instructor_id*}, {*instructor_id, instructor_name*} and {*instructor_name*} are the superkeys of *instructor*, if no two instructors have the same name
  - If tuples $t_1 \neq t_2$, then $t_1[K] \neq t_2[K]$



ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000

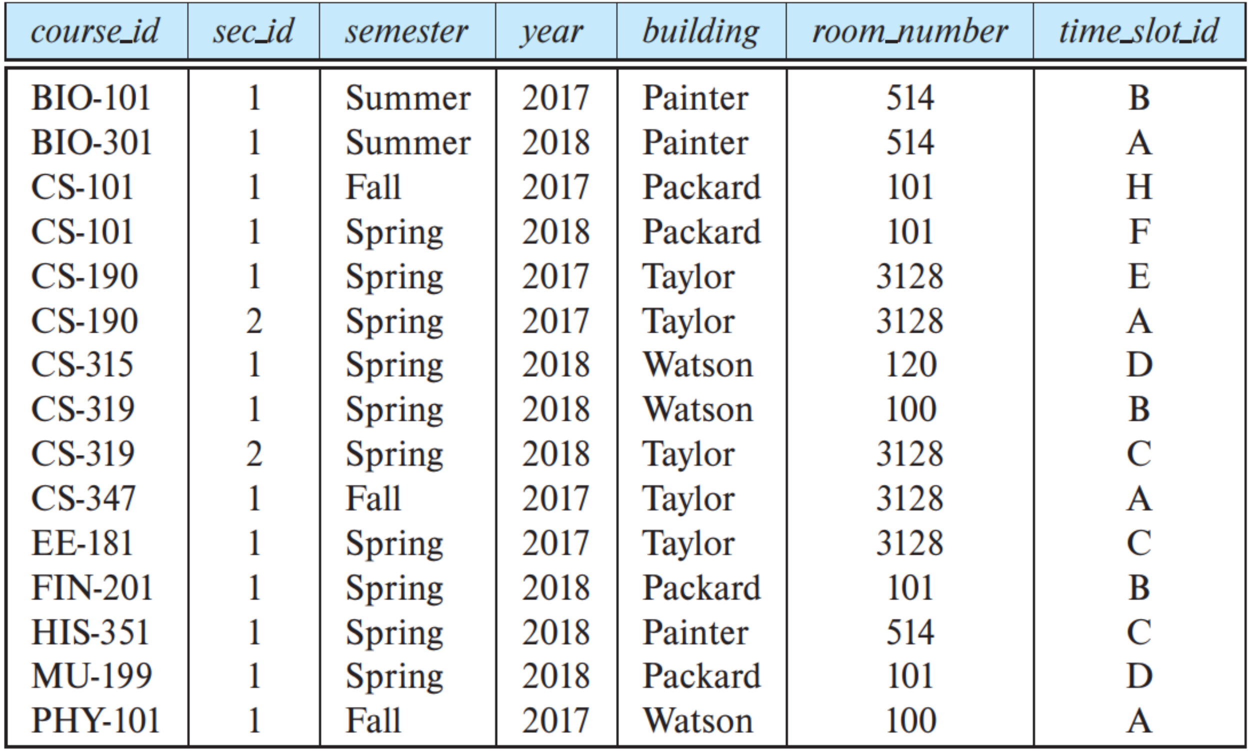
- **候选码(Candidate key)**
  - K is a candidate key if K is minimal
  - E.g., {*instructor_id*} is a candidate key for *instructor*
- **主码(Primary key)**
  - A candidate key is chosen by the DB designer to identify tuples within a relation

码(续)



思考：各个关系的候选码？

码(续)

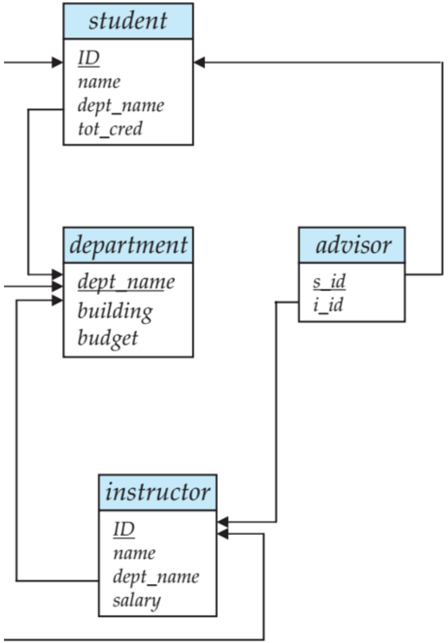


course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

section (course id, sec id, semester, year, building, room number, time slot id)

码(续)

- **外码(Foreign key)**
  - A relation schema R_1 may include among its attributes the primary key of another relation schema R_2. This attribute is called a foreign key from R_1, referencing R_2
  - Relation r_1 is called the referencing relation (参照关系) of the foreign key dependency, and r_2 is called the referenced relation (被参照关系) of the foreign key dependency

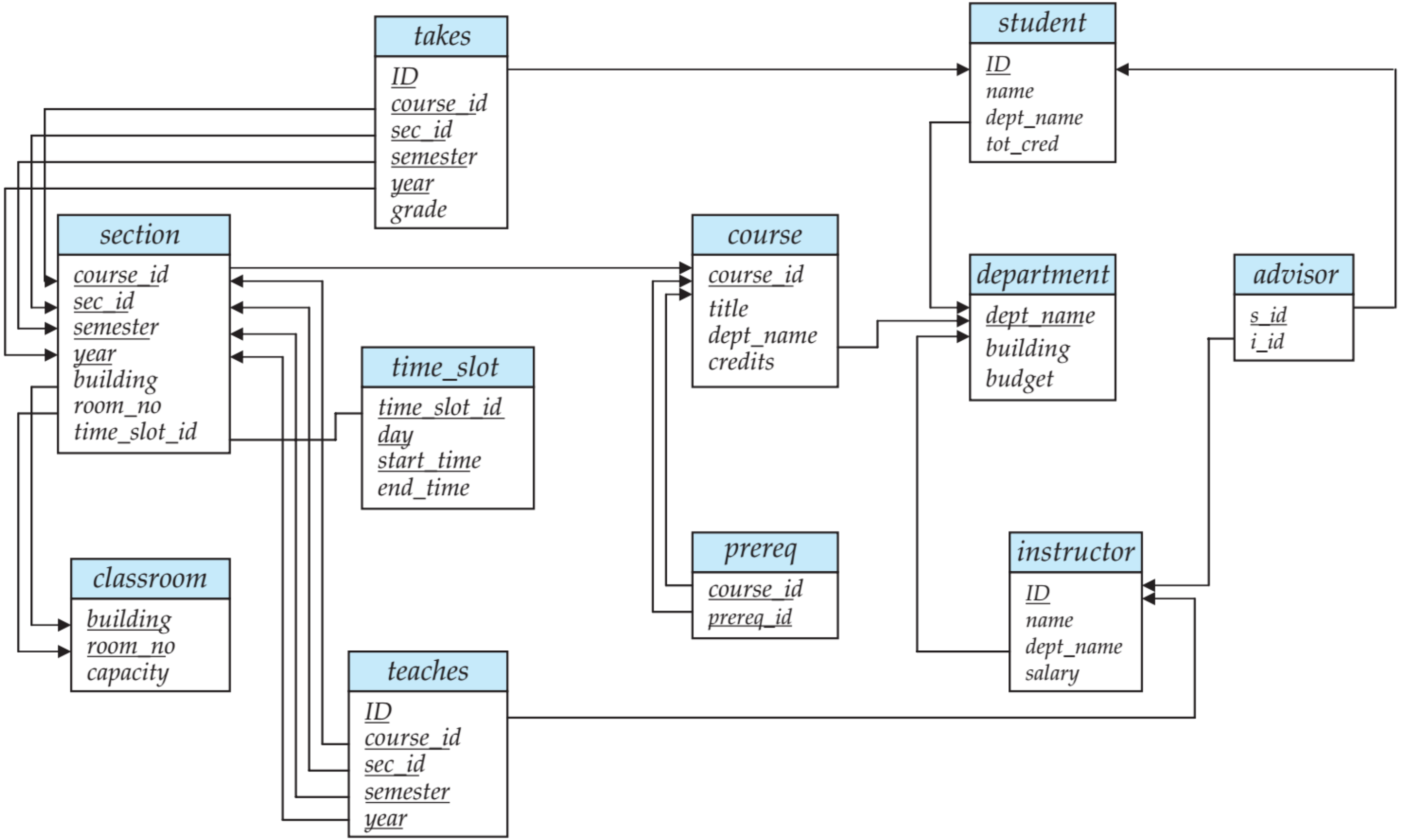


- **参照完整性约束(Referential integrity constraint)**
  - The values appearing in specified attributes of any tuple in the referencing relation should also appear in specified attributes of at least one tuple in the referenced relation

目录

- 关系数据库的结构
- 数据库模式
- 码
- **模式图**
- 关系查询语言
- 关系代数

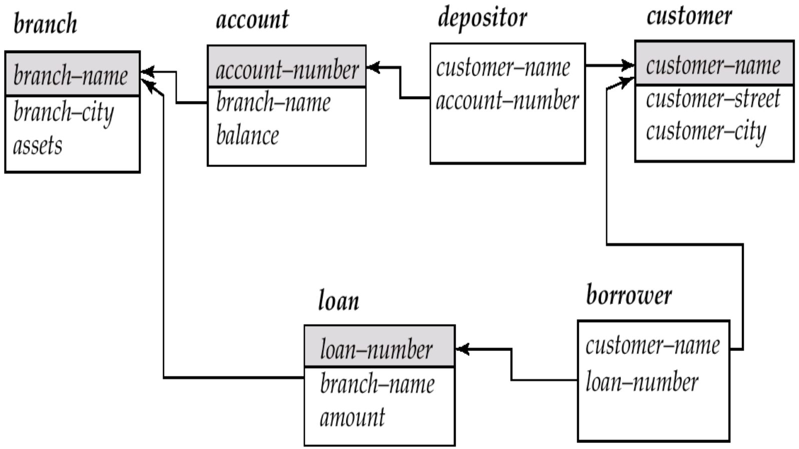
模式图(Schema Diagrams)



Schema diagram for the university database

模式图(Schema Diagrams)

Schema Diagram for the banking database



Relations in banking database

branch (branch_name, branch_city, assets)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)

depositor (customer_name, account_number)

borrower (customer_name, loan_number)

目录

- 关系数据库的结构
- 数据库模式
- 码
- 模式图
- **关系查询语言**
- 关系代数

关系查询语言

- **Relational query language**: used to request information from the database
- **Categories of query languages**
  - Procedural (过程式)/函数式
    - **Relational Algebra（关系代数）**
  - Non-procedural/Declarative （非过程式/声明式)
    - SQL（结构化查询语言）
    - Tuple Relational Calculus*（元组关系演算）
    - Domain Relational Calculus*（域关系演算）

目录

- 关系数据库的结构
- 数据库模式
- 码
- 模式图
- 关系查询语言
- **关系代数**

关系代数(Relational Algebra)

- A procedural language consisting of a set of operations that take one or two relations as input and produce a new relation as the result

- **Six basic operations（红色：一元运算，其他：二元运算)**
  - Select (选择)
  - Project (投影)
  - Union (集合并)
  - Set difference (集合差)
  - Cartesian product (笛卡尔积)
  - Rename (重命名)

选择操作(Select Operation)

- **Notation:** $\sigma_P(r) = \{t | t \in r \ and \ P(t)\}$
  - P is the selection predicate(选择谓词） consisting of ∧(and), ∨(or), ¬(not), =, ≠, <, >, ≤, ≥
  - 例:

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

relation r

A	B	C	D
α	α	1	7
β	β	23	10

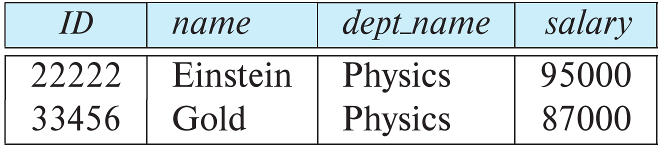
$\sigma_{A=B \land D>5}(r)$

选择操作(续)

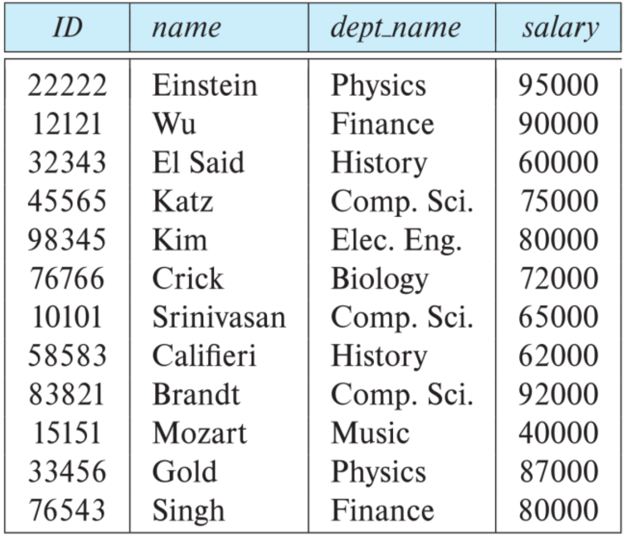
- 例：select those tuples of the instructor relation where the instructor is in the “Physics” department
  - **Query**

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

  - **Result**



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



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

Instructor relation

选择操作(续)

- Can combine several predicates into a larger predicate by using the connectives: $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)
  - 例：find the instructors with a salary greater $90,000 in Physics Department

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

- The select predicate may include comparisons between two attributes.
  - 例：find all departments whose names are the same as their building names:

$$\sigma_{dept\_name=building}\ (department)$$

投影操作(Project Operation)

- **符号:** $\Pi_{A_1,A_2,\ldots,A_k}(r)$
  - $A_1, A_2, \ldots, A_k$ are attribute names and r is a relation name
  - The result is defined as the relation of k columns obtained by selecting the columns that are listed
  - Duplicate rows are **removed** from the result, since relations are sets
  - 例:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

relation r

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

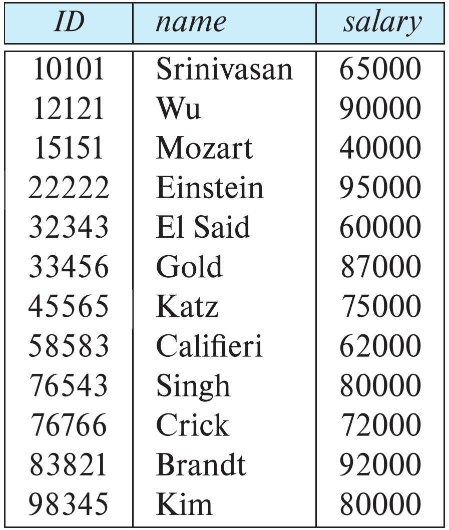
$\Pi_{A,C}(r)$

投影操作(续)

- 例：List the ID, name and salary of each *instructor*
- Query:

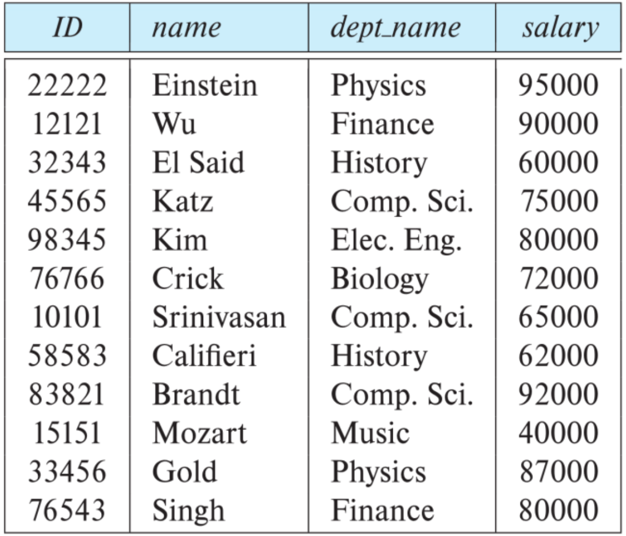
$$\Pi_{ID,\ name,\ salary}\ (instructor)$$

- Result:



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

思考：需要去重复元组吗?



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

Instructor relation

广义投影(Generalized Projection)

- Extend the projection operation by allowing arithmetic functions in the projection list $\Pi_{F_1,F_2,\ldots,F_n}(E)$
  - E is any relational-algebra expression
  - Each of $F_1, F_2, \ldots, F_n$ is a arithmetic expression involving constants and attributes in the schema of E

- Given relation *credit_info(customer_name, limit, credit_balance)*, find how much more each person can spend:
  - $\Pi_{customer\_name,\, limit-credit\_balance}(credit\_info)$

集合并(Set Union)

- **Notation:** $r \cup s = \{t | t \in r \text{ or } t \in s\}$
  - r, s must have the same arity (同元的), i.e., the same number of attributes
  - The attribute domains must be compatible (相容的)
    - E.g., the 2nd column of r has the same type of values as the 2nd column of s
  - 例：find all courses in the Fall 2022 semester, or in the Spring 2023 semester, or in both:

$$\Pi_{course\_id}(\sigma_{semester=\text{"Fall"}\ \wedge\ year=2022}(section)) \cup \Pi_{course\_id}(\sigma_{semester=\text{"Spring"}\ \wedge\ year=2023}(section))$$

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

relations r, s

A	B
α	1
α	2
β	1
β	3

$r \cup s$

集合差(Set Difference)

- **Notation:** $r - s = \{t | t \in r \; and \; t \notin s\}$
  - Set difference must be taken between compatible relations, i.e., r and s must have the same arity and attribute domains
  - 例:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

relations r, s

A	B
α	1
β	1

$r - s$

集合差(续)

- 例：find all courses taught in the Fall 2021 semester, but not in the Spring 2022 semester

$$\Pi_{course\_id}(\sigma_{semester=\text{“Fall”} \wedge\ year=2021}\ (section)) - \Pi_{course\_id}\ (\sigma_{semester=\text{“Spring”} \wedge\ year=2022}\ (section))$$

笛卡尔积(Cartesian Product)

- **Notation:** $r \times s = \{tq | t \in r \; and \; q \in s\}$
  - The attributes of $r(R)$ and $s(S)$ should be disjoint, i.e., $R \cap S = \emptyset$
  - If the attributes of $r(R)$ and $s(S)$ are not disjoint, then **renaming** is required

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

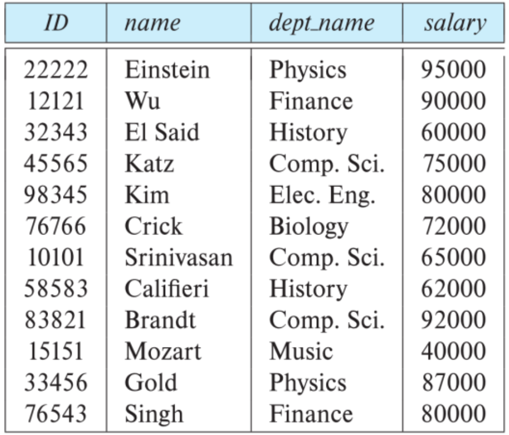
s

relations r, 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

$r \times s$

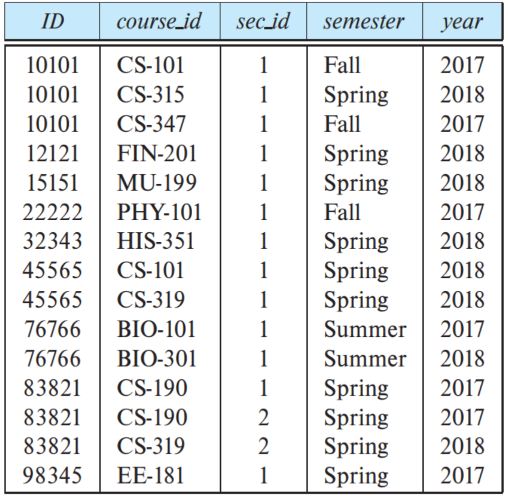
instructor



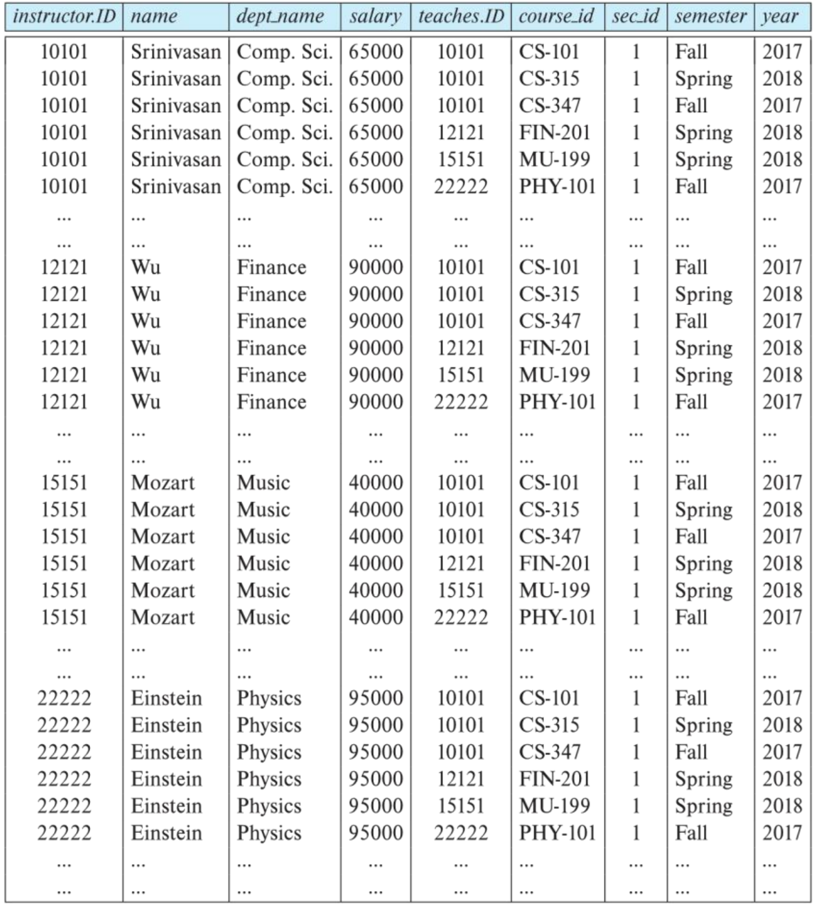
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

instructor x teaches

teaches



ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2017
10101	CS-315	1	Spring	2018
10101	CS-347	1	Fall	2017
12121	FIN-201	1	Spring	2018
15151	MU-199	1	Spring	2018
22222	PHY-101	1	Fall	2017
32343	HIS-351	1	Spring	2018
45565	CS-101	1	Spring	2018
45565	CS-319	1	Spring	2018
76766	BIO-101	1	Summer	2017
76766	BIO-301	1	Summer	2018
83821	CS-190	1	Spring	2017
83821	CS-190	2	Spring	2017
83821	CS-319	2	Spring	2018
98345	EE-181	1	Spring	2017



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
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...

组合操作

- Expressions using multiple operations
  - E.g., $\sigma_{A=C}(r \times s)$

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

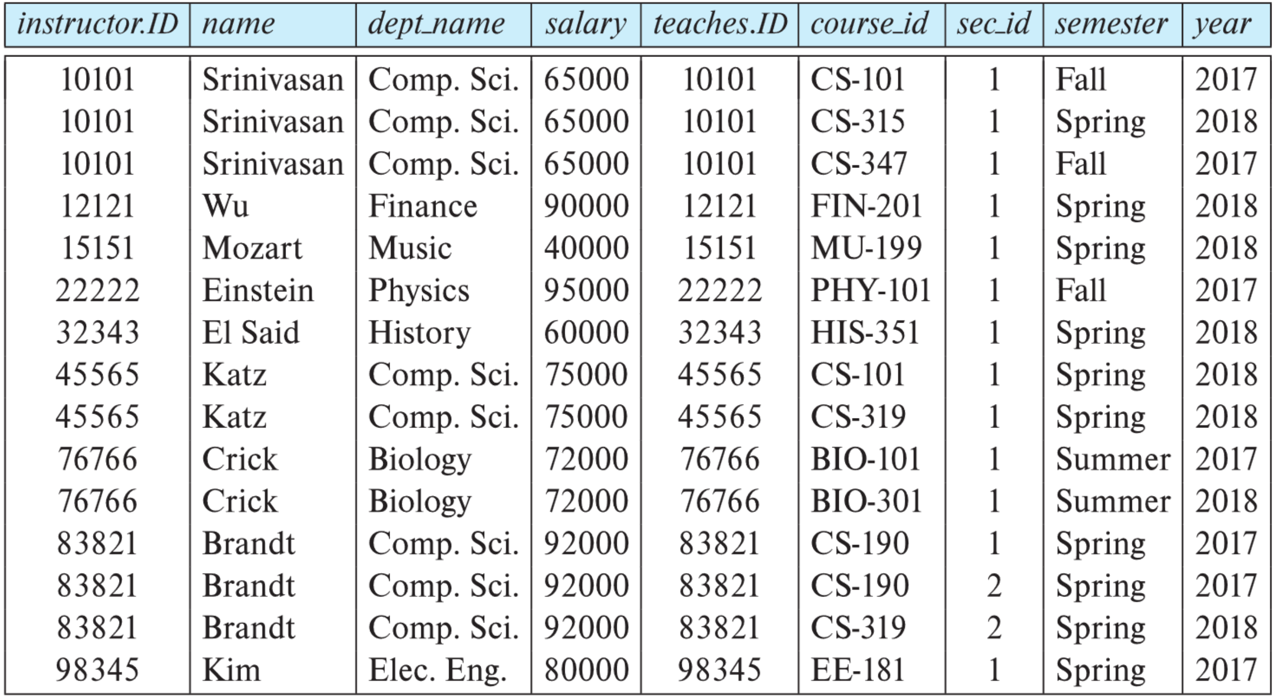
$r \times s$

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

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

组合操作(续)

- $\sigma_{instructor.id\ =\ teaches.id}$ *(instructor x 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

更名运算(Rename Operation)

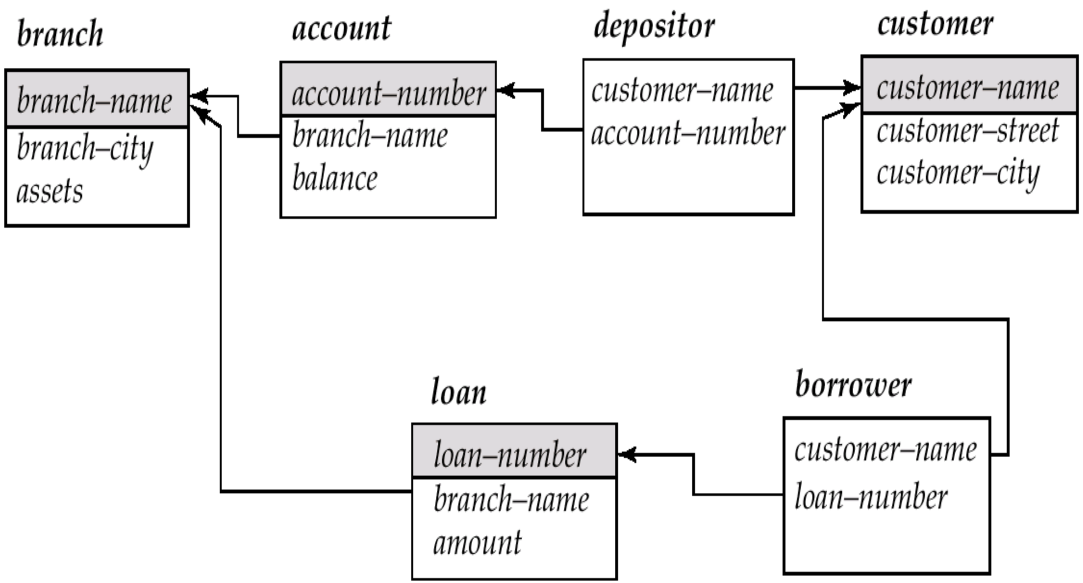
- Allows us to name, and therefore to refer to, the results of relational-algebra expressions
  - E.g., $\rho_X(E)$ returns the expression E under the name X

- If a relational-algebra expression E has n attributes
  - $\rho_{X(A_1,A_2,\ldots,A_n)}(E)$ returns the result of expression E under the name X, and with the attributes renamed to $A_1, A_2, \ldots, A_n$

关系查询语言注意事项

- 查询输入为单个或多个表格(关系)
- 查询输出为表格，且该表格中的数据都在输入表格中存在

后续示例所用关系数据库模式



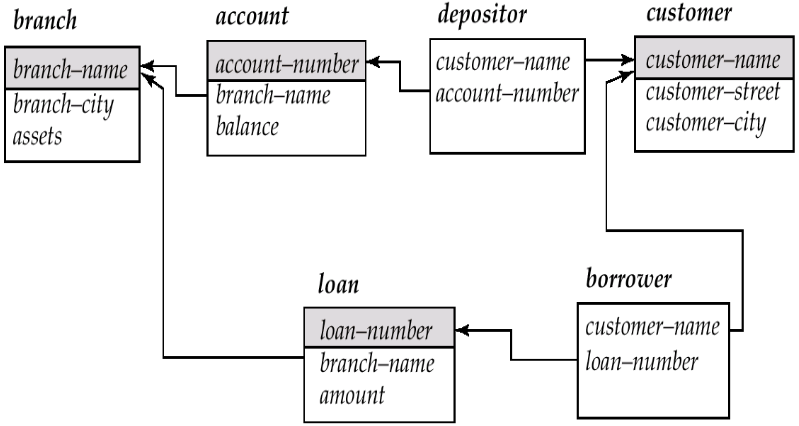
例1

- Find all loans of over $1200

$$\sigma_{amount>1200}(loan)$$

- Find the loan number for each loan of an amount greater than $1200

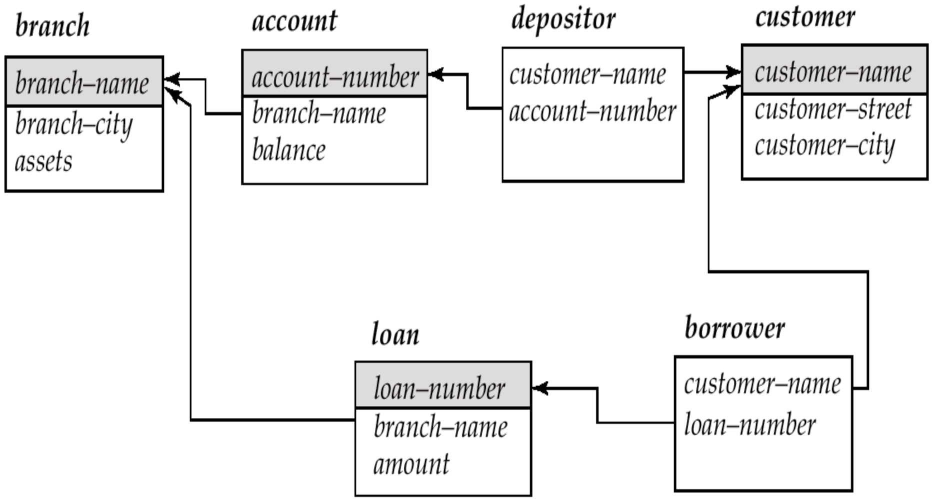
$$\Pi_{loan\_number}(\sigma_{amount>1200}(loan))$$



例2

- Find the names of all customers who have a loan, an account, or both, from the bank

$$\Pi_{customer\_name}(borrower) \cup \Pi_{customer\_name}(depositor)$$



例3

- Find the names of all customers who have a loan at the Jiading branch

$$\Pi_{customer\_name} (\sigma_{branch\_name=\text{“}Jiading\text{”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(borrower \times loan)))$$

- Find the names of all customers who have a loan at the Jiading branch but do not have an account at any branch of the bank

$$\Pi_{customer\_name} (\sigma_{branch\_name = \text{“}Jiading\text{”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(borrower \times loan))) - \Pi_{customer\_name}(depositor)$$

例4

- Find the names of all customers who have a loan at the Jiading branch
- **Query 1**

$$\Pi_{customer\_name}(\sigma_{branch\_name\ =\ \text{“Jiading”}}(\sigma_{borrower.loan\_number\ =\ loan.loan\_number}(borrower \times loan)))$$

- **Query 2**

$$\Pi_{customer\_name}(\sigma_{loan.loan\_number\ =\ borrower.loan\_number}((\sigma_{branch\_name\ =\ \text{“Jiading”}}(loan)) \times borrower))$$

思考：这两个查询表达式等价吗？它们有何区别?

例5

- Find the largest account balance
- **Stupid Strategy**
  - Rename account relation as d so that we can compare each account balance with all others
  - Find those balances that are not the largest
  - Use set difference to find those account balances that are not found in the previous steps

$$\Pi_{balance}(account) - \Pi_{account.balance}(\sigma_{account.balance < d.balance}\ (account \times \rho_{d}\ (account)))$$

关系表达式(Relational Expressions)

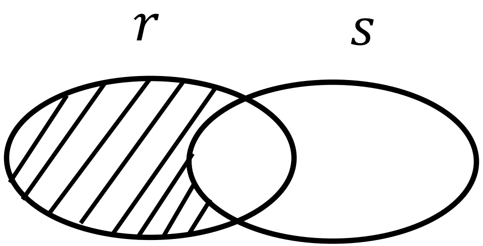
- A basic expression in the relational algebra consists of either one of the following:
  - A relation in the database
  - A constant relation, e.g., {(22222, Einsteir, Physics, 9500), (76543, Singh, Finance, 80000)}
- Let E_1 and E_2 be relational-algebra expressions, the following are all relational-algebra expressions
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - $E_1 \times E_2$
  - $\sigma_p(E_1)$, P is a predicate on attributes in E_1
  - $\Pi_s(E_1)$, S is a list of attributes in E_1
  - $\rho_X(E_1)$, X is the new name for the result of E_1

附加操作

- **附件操作：**用于简化查询书写，不额外增加关系代数的操作能力
  - Set intersection (集合交)
  - Natural join (自然连接)
  - Outer join（外连接）
  - Division (除)
  - Assignment (赋值)

集合交(Set Intersection)

- **Notation:** $r \cap s = \{t | t \in r \text{ and } t \in s\}$
  - r, s have the same number of attributes
  - the attributes of r and s are compatible
  - $r \cap s = r - (r - s)$



$r - s$

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

集合交(续)

- 例：find all the courses taught in both the Fall 2021 and the Spring 2022 semesters

$$\Pi_{course\_id}\ (\sigma_{semester=\text{“Fall”}\ \land\ year=2021}\ (section))\ \cap\ \Pi_{course\_id}\ (\sigma_{\ semester=\text{“Spring”}\ \land\ year=2022}\ (section))$$

自然连接(Natural Join)

- **Notation:** $r \bowtie s$
- Let r and s be the relations on schemas R and S, respectively. Then $r \bowtie s$ is a relation on schema $R \cup S$ obtained as follows
  - Consider each pair of tuples t_r from r and t_s from s
  - If t_r and t_s have the same value on each of the attributes in $R \cap S$, add a tuple t to the result, where
    - t has the same value as t_r on r
    - t has the same value as t_s on s
- 例：$R = (A,\ B,\ C,\ D), S = (E, B, D)$
  - Result schema: (A, B, C, D, E)
  - $r \bowtie s$ is defined as: $\Pi_{r.A,\, r.B,\, r.C,\, r.D,\, s.E}(\sigma_{r.B=s.B\ \wedge r.D=s.D}(r \times s))$

自然连接(续)

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

$r \bowtie s$:

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

自然连接(续)

- Let $r(R)$ and $s(S)$ be relations without any attributes in common, i.e., $R \cap S = \emptyset$. Then, $r \bowtie s = r \times s$
- ***θ*-join**
  - An extension to the natural-join operation that allows us to combine a selection and a **Cartesian product** into a single operation
  - Consider relations $r(R)$ and $s(S)$, and let θ be a predicate on attributes in the schema $R \cup S$. The theta join $r \bowtie_\theta s$ is defined as follows: $r \bowtie_\theta s = \sigma_\theta(r \times s)$

自然连接(续)

Relations r, s:

r:

A	B	C
a1	b1	5
a1	b2	6
a2	b3	8
a2	b4	12

s:

B	E
b1	3
b2	7
b3	10
b3	2
b5	2

A	B	C	E
a1	b1	5	3
a1	b2	6	7
a2	b3	8	10
a2	b3	8	2

$r \bowtie s$

A	R.B	C	S.B	E
a1	b1	5	b1	3
a1	b2	6	b2	7
a2	b3	8	b3	10
a2	b3	8	b3	2

$r \bowtie_{r.B=s.B} s$

A	R.B	C	S.B	E
a1	b1	5	b2	7
a1	b1	5	b3	10
a1	b2	6	b2	7
a1	b2	6	b3	10
a2	b3	8	b3	10

$r \bowtie_{C<E} s$

外连接(Outer Join)

- An extension of the join operation that avoids loss of information
- Compute the join and then add tuples from one relation that does not match tuples in the other relation to the join result
- Use null values:
  - null signifies that the value is unknown or does not exist
  - All comparisons involving null are (roughly speaking) false by definition

外连接(续)

loan-number	*branch-name*	*amount*
L-170 L-230 L-260	Downtown Redwood Perryridge	3000 4000 1700

Relation *loan*

customer-name	*loan-number*
Jones Smith Hayes	L-170 L-230 L-155

Relation *borrower*

loan-number	*branch-name*	*amount*	*customer-name*
L-170 L-230	Downtown Redwood	3000 4000	Jones Smith

Inner Join: *loan* ⋈ *Borrower*

loan-number	*branch-name*	*amount*	*customer-name*
L-170 L-230 L-260	Downtown Redwood Perryridge	3000 4000 1700	Jones Smith *null*

Left Outer Join: *loan* ⟕ *Borrower*

loan-number	*branch-name*	*amount*	*customer-name*
L-170 L-230 L-155	Downtown Redwood *null*	3000 4000 *null*	Jones Smith Hayes

Right Outer Join: *loan* ⟖ *borrower*

loan-number	*branch-name*	*amount*	*customer-name*
L-170 L-230 L-260 L-155	Downtown Redwood Perryridge *null*	3000 4000 1700 *null*	Jones Smith *null* Hayes

Full Outer Join: *loan* ⟗ *borrower*

除(Division)

- **Notation:** $r \div s$
  - r and s are relations on schemas R and S, respectively
    - $R = (A_1, \dots, A_m, B_1, \dots, B_n)$
    - $S = (B_1, \dots, B_n)$
  - The result of $r \div s$ is a relation on schema $R - S = (A_1, \dots, A_m)$, i.e.,

$$r \div s = \{t | t \in \Pi_{R-S}(r) \land \forall u \in s (tu \in r)\}$$

- A tuple t is in $r \div s$ if and only if both conditions hold:
  - t is in $\Pi_{R-S}(r)$
  - For every tuple t_s in s, there is a tuple t_r in r satisfying:
    - $t_r[S] = t_s[S]$
    - $t_r[R - S] = t$

除(续)

Relations *r*, *s* :

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
$\in$	6
$\in$	1
β	2

r

B
1
2

s

$r \div s$:

A
α
β

除(续)

- Definition in terms of the basic algebra operation
  - Let $r(R)$ and $s(S)$ be relations, and let $S \subseteq R$
  - $r \div s = \Pi_{R-S}(r) - \Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$
- To see why
  - $\Pi_{R-S,S}(r)$ simply reorders attributes of r
  - $\Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S,S}(r))$ gives those tuples t in $\Pi_{R-S}(r)$ such that for some tuple $u \in s$, $tu \notin r$

赋值运算(Assignment)

- The assignment operation (←) provides a convenient way to express complex queries
  - Write query as a sequential program consisting of
    - a series of assignments
    - followed by an expression whose value is displayed as a result of the query
  - Assignment must be made to a temporary relation variable
- E.g., write $r \div s$ as:
  - $temp_1 \leftarrow \Pi_{R-S}(r)$
  - $temp_2 \leftarrow \Pi_{R-S}((temp_1 \times s) - \Pi_{R-S,S}(r))$
  - $result = temp_1 - temp_2$

例6

- Find all the customers who have accounts from both “Jiading” and “Pudong” branches
- **Query 1**
  - $\Pi_{CN}\left(\sigma_{BN="Jiading"}(depositor \ \bowtie \ account)\right) \cap \Pi_{CN}(\sigma_{BN="Pudong"}(depositor \ \bowtie \ account))$
  - where CN denotes customer_name and BN denotes branch_name
- **Query 2**
  - $\Pi_{customer\_name,branch\_name}(depositor \ \bowtie \ account) \div \rho_{temp(branch\_name)} (\{(“Jiading”), (“Pudong”)\})$
  - Query 2 uses a constant relation

例7

- Find all the customers who have an account at all branches located in Shanghai
  - $\Pi_{customer\_name,branch\_name}(depositor \bowtie account) \div \Pi_{branch\_name}(\sigma_{branch\_city="Shanghai"}(branch))$

聚合函数(Aggregate Functions)

- **聚合操作：**
  - **avg:** average value
  - **min:** minimum value
  - **max:** maximum value
  - **sum:** sum of values
  - **count:** number of values

- **关系代数中的聚合函数**
  - ${}_{G_1,G_1,\ldots,G_n}g_{F_1(A_1),F_2(A_2),\ldots,F_n(A_n)}(E)$
    - E is any relational-algebra expression
    - $G_1, G_2, \ldots, G_n$ is a list of attributes on which to group (can be empty)
    - Each F_i is an aggregate function
    - Each A_i is an attribute name

聚合函数(续)

Relation *r* :

A	B	C
α	α	7
α	β	7
β	β	3
β	β	10

$g_{\text{sum(c)}}$ (r) :

Sum(C)
27

Relation account grouped by *branch_name*:

branch_name	account_number	balance
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

$_{\textit{branch\_name}}\, g_{\textit{sum(balance)}}$ ***(account)***

branch_name	balance
Perryridge	1300
Brighton	1500
Redwood	700

聚合函数(续)

- The result of aggregation does not have a name
  - Can use rename operation to give it a name
  - For convenience, we permit renaming as part of aggregate operation

$$_{branch\_name}\mathcal{G}_{\mathbf{sum}(balance)\ \mathbf{as}\ sum\_balance}(account)$$

小结

- **关系数据库的结构**
  - 关系、属性
  - 关系模式、关系实例
- **数据库模式**
- **码**
  - 超码、候选码、主码
  - 外码、参照完整性约束
- **模式图**
- **关系查询语言**
  - 过程式（关系代数）、非过程式（SQL）
- **关系代数**
  - **基本操作**：选择、投影、集合并、集合差、笛卡尔积、重命名
  - **附加操作**：集合交、自然连接、外连接、除、赋值、聚合函数

作业

- **Exercises (原书第7版)**
  - 2.1, 2.6, 2.7, 2.8, 2.15 （选择其中3个完成）
- **Submission**
  - Canvas上提交，上传单个PDF文件
  - Deadline: 待定