

Module 09

Partha Pratim Das

Objectives & Outline

Additional Basic Operations

Cartesian Product

Danama AS

String Values

Order By Clause

Clause
Where Clause

Predicates

Duplicates

Module Summar

Database Management Systems

Module 09: Introduction to SQL/2

Partha Pratim Das

Department of Computer Science and Engineering Indian Institute of Technology, Kharagpur

ppd@cse.iitkgp.ac.in

Module Recap

Module 09

Partha Pratii Das

Objectives & Outline

Additional Bas Operations

Cartesian Produc

Rename AS Operation

Order By Clause

Clause
Where Clause

Module Summar

- Introduced relational query language
- Familiarized with data definition and basic query structure

Module Objectives

Module 09

Partha Pratin Das

Objectives & Outline

Additional Bas Operations

Cartesian Product

Rename AS

String Values

Order By Clause Select Top / Fet

Where Clause Predicates

Module Summar

• To complete the understanding of basic query structure

Module Outline

Module 09

Partha Pratin Das

Objectives & Outline

Additional Basic Operations

Cartesian Produc

String Values
Order By Clause

Where Clause Predicates

Module Summai

• Additional Basic Operations

- Cartesian Product
- Rename AS Operation
- String Values
- Order By
- Select Top / Fetch
- Where Clause Predicate
- Duplicates

Additional Basic Operations

Module 09

Partha Pratio

Objectives Outline

Additional Basic Operations

Cartesian Product

Rename AS

String Values

Order By Clause Select Top / Feto

Where Clause Predicates

Module Summar

Additional Basic Operations

Database Management Systems Partha Pratim Das 09.5



Cartesian Product

Module 09

Partha Pratir Das

Objectives Outline

Additional Basic Operations Cartesian Product Rename AS

String Values Order By Clause Select Top / Fetch Clause Where Clause Predicates

Module Summar

• Find the Cartesian product *instructor X teaches*

select *

from instructor, teaches

- generates every possible instructor-teaches pair, with all attributes from both relations
- For common attributes (for example, *ID*), the attributes in the resulting table are renamed using the relation name (for example, *instructor.ID*)
- Cartesian product not very useful directly, but useful combined with where-clause condition (selection operation in relational algebra)



Cartesian Product

Module 09

Partha Pratir Das

Objectives Outline

Additional Bas Operations

Cartesian Product

- Cartesian Froduct

String Values Order By Clause

Select Top / Feto Clause

Where Clause Predicates

Module Summar

instructor teaches course id sec id semester name dept name salary year 10101 Sriniyasan Comp. Sci. 65000 10101 CS-101 Fall 2009 12121 Wii Finance 90000 10101 CS-315 2010 Spring 15151 Mozart Music 40000 10101 CS-347 2009 Fall 22222 Einstein 95000 Physics 12121 FIN-201 Spring 2010 32343 El Said 60000 History MII-199 2010 15151 Spring 33456 Gold Physics 87000 PHY-101 2009 Fall 45565 Katz Comp. Sci. 75000 32343 HIS-351 Spring 2010 58583 Califieri 62000 History 45565 CS-101 2010 Spring 76543 Singh Finance 80000 45565 CS-319 Spring 2010 76766 72000 Crick Biology 76766 BIO-101 Summer 2009 83821 2010 Inst.ID dept_name salary teaches.ID course id sec id semester name year 98345 2009 Srinivasan Comp. Sci. 65000 10101 CS-101 Fall 2009 2009 10101 Srinivasan Comp. Sci. 65000 10101 CS-315 2010 10101 Spring 2010 10101 Srinivasan Comp. Sci. 65000 10101 CS-347 Fall 2009 2009 2010 Sriniyasan Comp. Sci. 65000 12121 FIN-201 Spring Srinivasan Comp. Sci. 65000 15151 MU-199 Spring 2010 10101 Srinivasan Comp. Sci. 65000 10101 22222 PHY-101 Fall 2009 Wu Finance 90000 10101 CS-101 Fall 2009 Wu Finance 10101 CS-315 Spring 2010 90000 Wu 10101 CS-347 Fall 2009 12121 Pinance 90000 FIN-201 2010 12121 Wu Pinance 90000 12121 Spring 12121 Wu 15151 MU-199 Spring 2010 Finance 90000 12121 PHY-101 2009 Wu Pinance 90000 Fall

Module 09

Partha Pratio

Objectives Outline

Additional Base
Operations
Cartesian Produc

Cartesian Product
Rename AS

String Values
Order By Clause
Select Top / Fetch
Clause
Where Clause

Module Summa

Find the names of all instructors who have taught some course and the course_id

select name, course_id

from instructor, teaches

where instructor.ID = teaches.ID

Equi-Join, Natural Join

instructor									teaches										
ID		name	name dept_name			пате	- 8	alary	1	ID		cours	e_jd	80	sec_id		semester		year
10101			rinivasan Comp. Sci			np. Sci.		65000	7 i	10101		CS-1	CS-101		1		Fall		2009
12121		Wu	Finance				90000		1010		CS-3	CS-315		1	Sı	oring	П	2010	
15151		Moza	Music				40000		10101		CS-3	CS-347		1 E		ill	П	2009	
22222		Einste	Physics				95000		12121		FIN-201			1		Spring		2010	
32343		El Sais	History				60000		15151		MU-199			1		Spring		2010	
33456		Gold	Physics				87000		22222		PHY-101					all		2009	
45565						np. Sci.		75000		32343		HIS-351		1		Spring			2010
58583			Califieri His							455		CS-1			1		oring		2010
76543		Singh		Finance				80000		455		CS-3			1	Spring			2010
76766		Crick		Biology			_	72000		76766		1 BIO-	BIO-101		1 1	Summer		!	2009
83821		Inst.ID	nan	ue .	П	dept_no	me	salary	teach	s.ID	α	ourse ic	sec.	id	seme	ster	year	П	2010
98345	5 4			-	-	-							1				_	H	2009
		10101	Srini	vas	an	Comp.	Sci.	65000	1010			5-101	1		Fall		2009	П	2009
		10101				Comp.			1010			5-315 5-347	1		Spri	ng	2010	П	2010
	- 1	10101				Comp.			1215			N-201	1 1				2009	IJ	2009
						Comp.			1515				1 1		Spri		2010	Г	
-	_	10101							2222			U-199 IV 101	1:		Spri	ng	2010		
		10101	SIM	Vas	nn	Comp.	oct.	60000			ľ		1 ^					г	
			100			111					ш		***	- 1	100		-0.0		
	- 1	12121	Win			Finance		90000	1010		Ы	 5-101		- 1	Fall		2009		
			Wu			Finance		90000	1010			3-101	1:		Spri		2010		
	_	12121	Wu			Pinano		90000	1010			5-347	1		Fall	ng	2009	Г	
	_		Wu			Pinano		90000	1212			N-201	1:		Spri		2010		-
		12121	Wu			Finance			1515			H-199	1		Spri		2010		
		12121	Wu			Pinance		90000	2222			U-199 IY-101	1		Fall	mg.	2010	г	
	╗				П		-			-	1.4		1 -					г	
		***				***													
	- 1					0.0		***	***					- 1				1	

Module 09

Partha Pratii Das

Objectives Outline

Additional Base

Cartesian Product

Panama AS

String Values
Order By Clause

Order By Clause Select Top / Fetcl Clause

Where Clause Predicates

Module Summar

• Find the names of all instructors in the Art department who have taught some course and the course_id

```
(select name, course_id
(from instructor, teaches
where instructor.ID = teaches.ID and instructor.dept_name = 'Art')
```



Rename AS Operation

Module 09

Partha Pratim Das

Objectives Outline

Additional Basic Operations Cartesian Product

Rename AS

String Values Order By Clause Select Top / Fetch Clause

Where Clause Predicates Duplicates

Module Summai

- The SQL allows renaming relations and attributes using the as clause:
 old_name as new_name
- Find the names of all instructors who have a higher salary than some instructor in 'Comp. Sci'.

select distinct T.name **from** instructor **as** T, instructor **as** S, **where** T.salary > S.salary **and** S.dept_name = 'Comp. Sci'

• Keyword **as** is optional and may be omitted

instructor as $T \equiv instructor T$



Cartesian Product Example

Module 09

Partha Pratir Das

Objectives Outline

Additional Bas Operations

Cartesian Product

Rename AS Operation String Values

Order By Clause Select Top / Fetcl Clause

Where Clause Predicates Duplicates

Module Summary

• Relation *emp_super*

person	supervisor
Bob	Alice
Mary	Susan
Alice	David
David	Mary

- Find the supervisor of "Bob"
- Find the supervisor of the supervisor of "Bob"
- Find ALL the supervisors (direct and indirect) of "Bob"



String Operations

Module 09

Partha Pratin Das

Objectives Outline

Operations

Cartesian Product

Rename AS

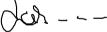
String Values Order By Clause

Select Top / Fetc Clause Where Clause Predicates

Module Summa

- SQL includes a string-matching operator for comparisons on character strings. The operator **like** uses patterns that are described using two special characters:
 - o percent (%). The % character matches any substring
 - \circ underscore ($_$). The $_$ character matches any character
- Find the names of all instructors whose name includes the substring "dar" select name
 - from instructor
 where name like '%dar%'
- Match the string "100%" like '100%' escape '\'
- in that above we use backslash (\) as the escape character







String Operations (2)

Module 09

Partha Pratin Das

Objectives a Outline

Operations

Cartesian Product

Rename AS

String Values
Order By Clause
Select Top / Fet

Select Top / Feto Clause Where Clause Predicates Duplicates

Module Summar

- Patterns are case sensitive
- Pattern matching examples:
 - 'Intro%' matches any string beginning with "Intro"
 - '%Comp%' matches any string containing "Comp" as a substring
 - '_ _ _' matches any string of exactly three characters
 - o '_ _ _ %' matches any string of at least three characters
- SQL supports a variety of string operations such as
 - concatenation (using "|")
 - converting from upper to lower case (and vice versa)
 - finding string length, extracting substrings, etc.



Ordering the Display of Tuples

Module 09

Partha Pratin Das

Objectives Outline

Additional Basic Operations Cartesian Product Rename AS Operation String Values

Order By Clause
Select Top / Fetch
Clause
Where Clause

Module Summar

• List in alphabetic order the names of all instructors

select distinct name from instructor

order by name

 We may specify desc for descending order or asc for ascending order, for each attribute; ascending order is the default.

Example: order by name desc

Can sort on multiple attributes

Example: order by dept_name, name



Selecting Number of Tuples in Output

Module 09

Partha Pratin Das

Objectives Outline

Additional Basic Operations Cartesian Product Rename AS Operation String Values

Order By Clause Select Top / Fetch Clause

Where Clause Predicates Duplicates

Module Summai

- The **Select Top** clause is used to specify the number of records to return
- The **Select Top** clause is useful on large tables with thousands of records. Returning a large number of records can impact performance

select top 10 distinct name from instructor

- from instructor
- Not all database systems support the SELECT TOP clause.
 - SQL Server & MS Access support select top
 - MySQL supports the **limit** clause
 - Oracle uses fetch first n rows only and rownum

select distinct name from instructor order by name fetch first 10 rows only



Where Clause Predicates

Module 09

Partha Pratin Das

Objectives Outline

Operations

Cartesian Product

String Values
Order By Clause
Select Top / Fetch

Where Clause Predicates Duplicates

Module Summai

- SQL includes a between comparison operator
- Example: Find the names of all instructors with salary between \$90,000 and \$100,000 (that is, \geq \$90,000 and \leq \$100,000)

select name from instructor

where salary between 90000 and 100000

Tuple comparison

select name, course_id

from *instructor*, *teaches*

where (instructor.ID, dept_name) = (teaches.ID, 'Biology');



In Operator

Module 09

Partha Pratin Das

Objectives Outline

Additional Bas Operations Cartesian Produc Rename AS Operation

String Values
Order By Clause
Select Top / Feto

Where Clause Predicates

Module Summa

- The in operator allows you to specify multiple values in a where clause
- The in operator is a shorthand for multiple or conditions
 - select name
 - from instructor
 - where dept_name in ('Comp. Sci.', 'Biology')



Duplicates

Module 09

Partha Pratin Das

Objectives Outline

Operations

Cartesian Product

Rename AS

Operation

String Values
Order By Clause
Select Top / Fetch
Clause
Where Clause

DuplicatesModule Summar

- In relations with duplicates, SQL can define how many copies of tuples appear in the result
- Multiset versions of some of the relational algebra operators given multiset relations r_1 and r_2 :
 - a) $\sigma_{\theta}(r_1)$: If there are c_1 copies of tuple t_1 in r_1 , and t_1 satisfies selections σ_{θ} , then there are c_1 copies of t_1 in $\sigma_{\theta}(r_1)$
 - b) $\Pi_A(r)$: For each copy of tuple t_1 in r_1 , there is a copy of tuple $\Pi_A(t_1)$ in $\Pi_A(r_1)$ where $\Pi_A(t_1)$ denotes the projection of the single tuple t_1
 - c) $r_1 \times r_2$: If there are c_1 copies of tuple t_1 in r_1 and c_2 copies of tuple t_2 in r_2 , there are $c_1 \times c_2$ copies of the tuple $t_1.t_2$ in $r_1 \times r_2$

Duplicates (2)

Module 09

Partha Pratim Das

Objectives Outline

Operations

Cartesian Product

Rename AS

Operation

String Values

String Values Order By Clause Select Top / Fetch Clause Where Clause

Where Clause Predicates Duplicates

Module Summary

• Example: Suppose multiset relations $r_1(A, B)$ and $r_2(C)$ are as follows: $r_1 = \{(1, a)(2, a)\}$ $r_2 = \{(2), (3), (3)\}$

• Then $\Pi_B(r_1)$ would be $\{(a), (a)\}$, while $\Pi_B(r_1) \times r_2$ would be $\{(a, 2), (a, 2), (a, 3), (a, 3), (a, 3), (a, 3)\}$

SQL duplicate semantics:

select
$$A_1, A_2, \dots, A_n$$

from r_1, r_2, \dots, r_m
where P

is equivalent to the *multiset* version of the expression:

$$\Pi_{A_1,A_2,\ldots,A_n}(\sigma_P(r_1\times r_2\times\ldots\times r_m))$$



Module Summary

Module 09

Partha Pratio

Objective: Outline

Operations

Cartesian Produc

Operation String Values Order By Clause Select Top / Fetch Clause

Where Clause Predicates Duplicates

Module Summary

• Completed the understanding of basic query structure

Slides used in this presentation are borrowed from http://db-book.com/ with kind permission of the authors.

Edited and new slides are marked with "PPD".