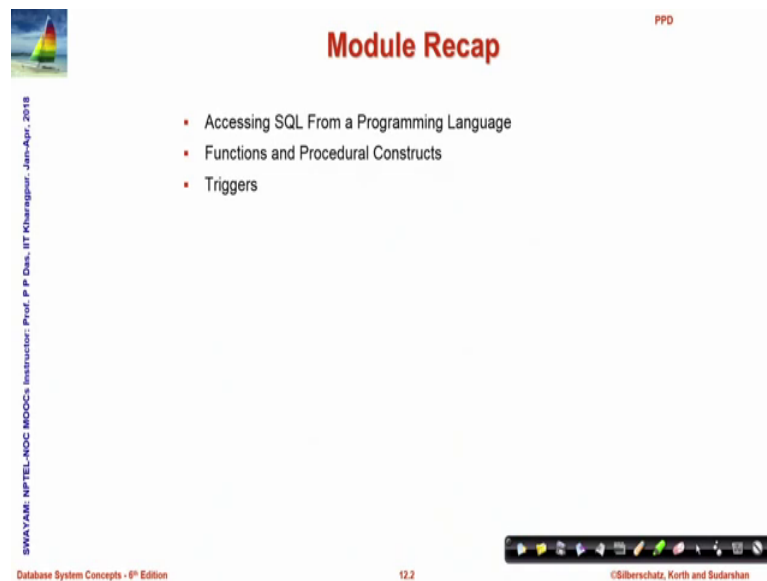


Database Management System
Prof. Partha Pratim Das
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture - 12
Formal Relational Query Languages

Welcome to module twelve of database management systems, in this module we will talk about the formal relational query languages. In the last couple of modules we have discussed about SQL at length introducing it dealing with the intermediate level of SQL features, and then exposing to some of the advanced features as well. The foundational mathematical model of SQL the query languages are to be discussed in this present module.

(Refer Slide Time: 01:04)



The slide is titled "Module Recap" in red text. It features a bulleted list of topics: "Accessing SQL From a Programming Language", "Functions and Procedural Constructs", and "Triggers". The slide is part of a presentation by Prof. P. P. Das, IIT Kharagpur, as indicated by the vertical text on the left. The bottom of the slide shows the slide number "12.2" and the copyright notice "©Silberschatz, Korth and Sudarshan".

PPD

Module Recap

- Accessing SQL From a Programming Language
- Functions and Procedural Constructs
- Triggers

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

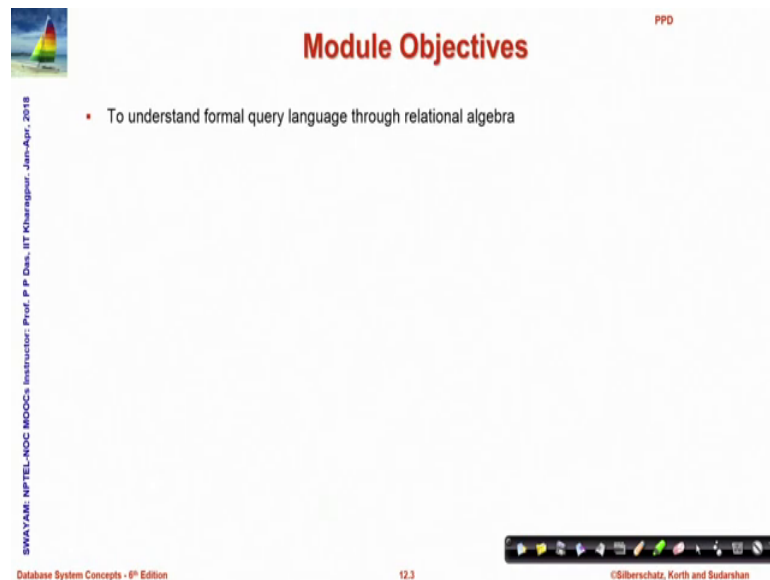
Database System Concepts - 8th Edition

12.2

©Silberschatz, Korth and Sudarshan

So, this is what we had done in the last module.

(Refer Slide Time: 01:10)



Module Objectives

- To understand formal query language through relational algebra

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition

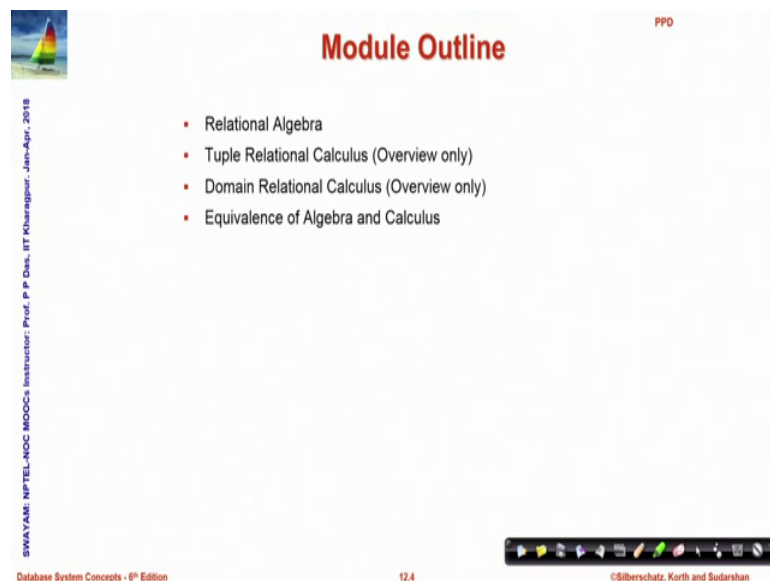
12.3

©Silberschatz, Korth and Sudarshan

PPD

In the current 1 we will work to understand the formal query languages.

(Refer Slide Time: 01:19)



Module Outline

- Relational Algebra
- Tuple Relational Calculus (Overview only)
- Domain Relational Calculus (Overview only)
- Equivalence of Algebra and Calculus

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition

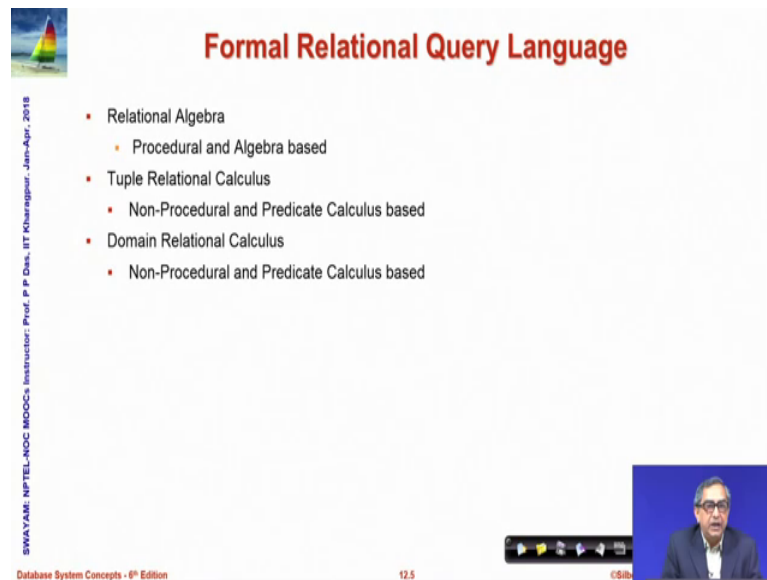
12.4

©Silberschatz, Korth and Sudarshan

PPD

Primarily through relational algebra, and then we will also take a look into some of the calculus aspects tuple relational calculus, and domain relational calculus and we will show by example the equivalence between the algebra and the 2 calculus.

(Refer Slide Time: 01:40)



Formal Relational Query Language

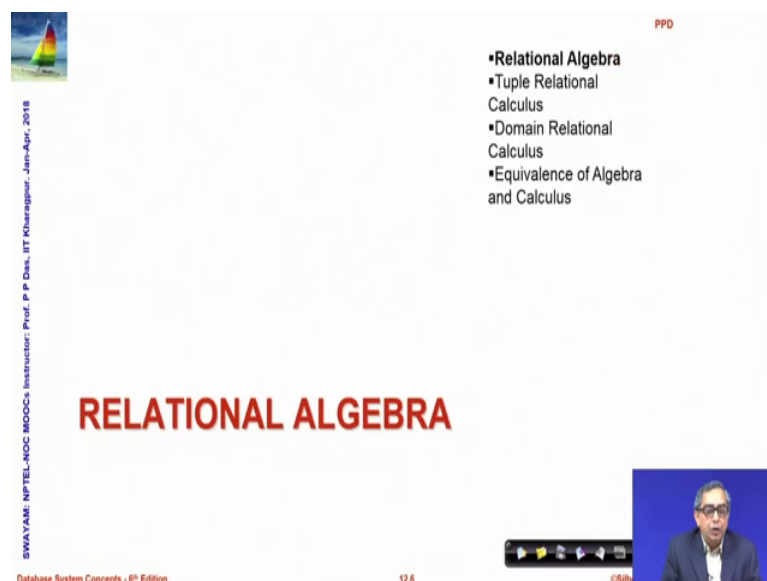
- Relational Algebra
 - Procedural and Algebra based
- Tuple Relational Calculus
 - Non-Procedural and Predicate Calculus based
- Domain Relational Calculus
 - Non-Procedural and Predicate Calculus based

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.5

So, formal relational query languages are of 3 types 1 is known as relational algebra this is procedural in nature. So, we specify what operations need to be done to achieve the result and the whole formulation is based on set algebra. The second formal query language is tuple relational calculus which is non procedural and is based on predicate calculus. The third one the domain relational calculus is a minor variant of the people relational calculus is and is also non procedural and predicate calculus based.

(Refer Slide Time: 02:35)



RELATIONAL ALGEBRA

- Relational Algebra
- Tuple Relational Calculus
- Domain Relational Calculus
- Equivalence of Algebra and Calculus

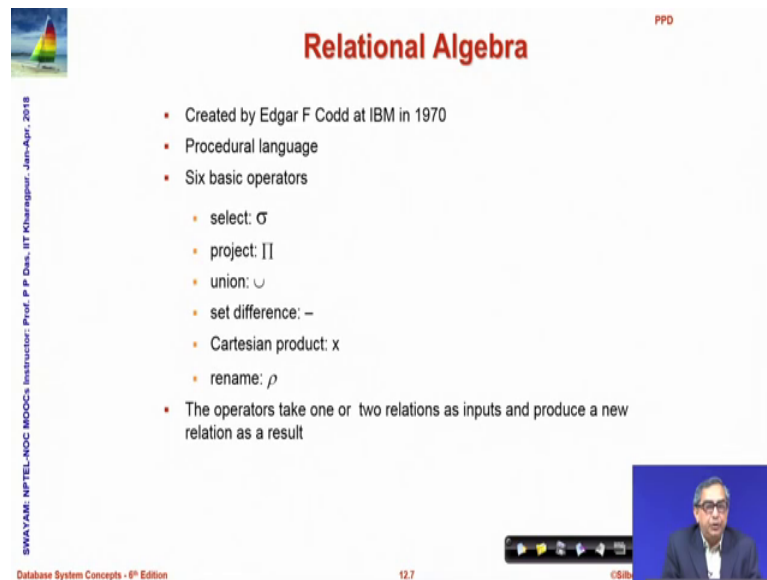
PPD

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.6

So, we start with the relational algebra in the relational algebra.

(Refer Slide Time: 02:40)




PPD

Relational Algebra

- Created by Edgar F Codd at IBM in 1970
- Procedural language
- Six basic operators
 - select: σ
 - project: Π
 - union: \cup
 - set difference: $-$
 - Cartesian product: \times
 - rename: ρ
- The operators take one or two relations as inputs and produce a new relation as a result

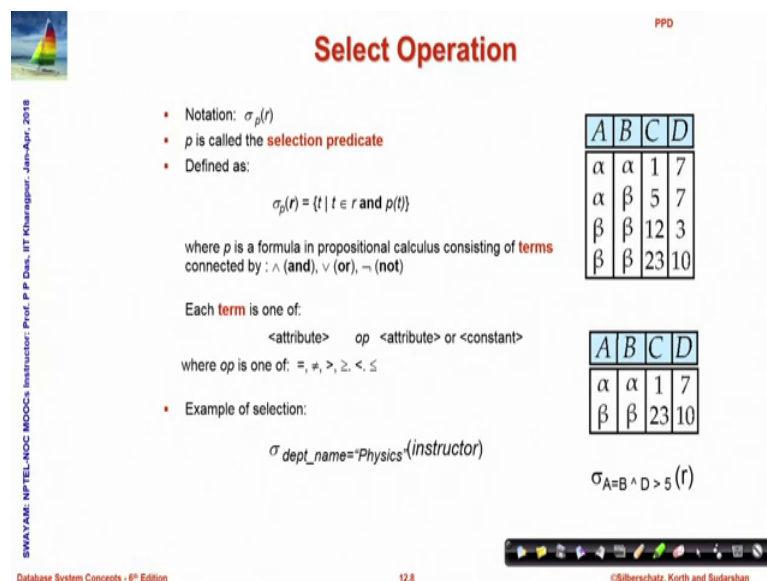
SWAYAM: NPTEL-NOC MOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 6th Edition 12.7 ©Silberschatz, Korth and Sudarshan



It was created by Edgar F Codd at IBM in 1970. So, you can see that it is quite an old formulation it is a procedural language it has six operators we have taken a quick view of these earlier in this module we will look at them at length. The select project union set difference Cartesian product and rename, we will also look at few derived operations like intersection and division which can be expressed in terms of these basic operators. And each one of these operators can take 1 or 2 relations as input and they produce 1 relation as a result.

(Refer Slide Time: 03:35)



PPD

Select Operation

- Notation: $\sigma_p(r)$
- p is called the **selection predicate**
- Defined as:
$$\sigma_p(r) = \{t \mid t \in r \text{ and } p(t)\}$$
where p is a formula in propositional calculus consisting of **terms** connected by : \wedge (**and**), \vee (**or**), \neg (**not**)
- Each **term** is one of:
$$\langle \text{attribute} \rangle \quad \text{op} \quad \langle \text{attribute} \rangle \text{ or } \langle \text{constant} \rangle$$
where op is one of: $=, \neq, >, \geq, <, \leq$
- Example of selection:
$$\sigma_{\text{dept_name}=\text{'Physics'}}(\text{instructor})$$

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

A	B	C	D
α	α	1	7
β	β	23	10

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

SWAYAM: NPTEL-NOC MOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 6th Edition 12.8 ©Silberschatz, Korth and Sudarshan

So, we start with the select operation which you know has a notation σ_p where p is a predicate it is called the selection predicate and within parentheses we have a relation r on which this predicate applies. So, it is defined as a set where you collect all the tuples all the rows designated by t and you specify that t belongs to the relation. So, it already exists in that relation and it satisfies the particular selection predicate.

So, any tuple that satisfies this predicate is included in the result any that does not satisfy is excluded from the result, p here particularly is a propositional calculus formula or expression. Where we have different terms that are connected by conjunction or and disjunction or negation or not, and each term by itself could be something like this it is an attribute operator and an attribute where operators are different comparisons operators 1 of the any six or a term could be an attribute operator a constant.

So, given that we can write any expression, which is a predicate and applying that we can select the tuples from the relation r which satisfy this predicate. So, here we show an simple example instructor is a relation department name is an attribute within course physics is a constant or literal. So, this selection show will select all the tuples where the attribute department name is equal to physics and all the others will be eliminated for reference I have also quoted here the example that we had shown at the time of introducing relational algebra.

So, you can see that here we have a more complex propositional term propositional formula where there are 2 terms, the a should equal b and d should be greater than 5. So, in the selection result both of these conditions must be satisfied by all the tuples which feature here. So, this is the first operation that relational algebra has the select operation.

(Refer Slide Time: 06:40)

Project Operation

Notation: $\Pi_{A_1, A_2, \dots, A_k}(r)$

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

- The result is defined as the relation of k columns obtained by erasing the columns that are not listed
- Duplicate rows removed from result, since relations are sets
- Example: To eliminate the *dept_name* attribute of *instructor*

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

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

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


Database System Concepts - 8th Edition 12.9 ©Silberschatz, Korth and Sudarshan

Next we move on to the second operation which is a project operation where a relation can be projected in terms of a number of attributes. So, π is the notation r again is the relation and the subscript at A_1, A_2, A_k , are key attribute names k has to be at least 1 and these attributes will be retained in the relation.

So, the result is defined as the relation of k columns by erasing all the columns of r which are not listed amongst this a $2 \times k$. Naturally, if you erase some columns it is possible that 2 rows that were distinct in those columns, but are identical in A_1 to A_k feature in the relation since every relation is a set no distinct no 2 copies of the same people are allowed. So, the duplicate rows will be removed from the result remind you this is in contrast to what sql does by default where duplicates or multi sets are allowed by default here we are talking about the formal relational algebra where it is purely set theoretic.

So, duplicate rows on projection will be removed from the result. So, we have an example from the instructor relation we had seen earlier we are projecting id name and salary. So, we are removing the department name, which also exists in the same relation and as a reference you can see the example that we had seen earlier while introducing the relational algebra where projection is done from 3 columns ABC into 2 columns A and C, and this results in at least results in 2 rows which are identical and therefore, in the final result 1 of those the duplicate 1 is removed.

(Refer Slide Time: 08:57)



Union Operation

PPD

- Notation: $r \cup s$
- Defined as:
$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$
- For $r \cup s$ to be valid.
 - r, s must have the same **arity** (same number of attributes)
 - 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 2009 semester, or in the Spring 2010 semester, or in both

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

A	B
α	1
α	2
β	1
β	3

$r \cup s$

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

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Khargpur, Jan-Apr, 2018

Database System Concepts - 8th Edition

12.10

©Silberschatz, Korth and Sudarshan

Moving on that the third operation is quite simple it is set theoretic union. So, our union s where r and s are 2 relations our set of peoples which either belong to r or belongs to s , or belongs to both, the condition is you can take union if both these relations have the same additive and the order of the attributes must satisfy that every corresponding attribute must have compatible domains. So, if we I talked about the second column of r , and if we talk about the second column of s they must be of the same type and this must hold for all columns that the union forms that is all columns of r as well as s , otherwise this operation is not defined.

So, as an example we show that to find all courses taught in fall 2009 semester, this is a this is the query where we do a selection to find all tuples which are taught, which represent courses taught in fall 2009 semester, from the section relation we do a projection to get the ids of those courses only. And the second row tells you the courses that are taught in the spring 2010 semester, and we do a union to get courses that are taught either in fall 2009 semester, or in spring 2010 semester, or both. This is how the union is performed and this is the earlier example repeated here.

(Refer Slide Time: 10:54)

PPD

Set Difference Operation

- Notation $r - s$
- Defined as:
$$r - s = \{t \mid t \in r \text{ and } t \notin s\}$$
- 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 2009 semester, but not in the Spring 2010 semester
$$\Pi_{\text{course_id}}(\sigma_{\text{semester}=\text{"Fall"} \wedge \text{year}=2009}(\text{section})) - \Pi_{\text{course_id}}(\sigma_{\text{semester}=\text{"Spring"} \wedge \text{year}=2010}(\text{section}))$$

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

A	B
α	1
β	1

$r - s$


SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Khargpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.11 ©Silberschatz, Korth and Sudarshan

set difference is again just simple difference of sets r minus s where a tuple belongs to r and it does not belong to s . So, you remove all the tuples belonging to s that exist in r to get r minus s again they must have the compatibility of having the same arity and attribute corresponding attribute domains must be compatible, this is an example to show to find all courses taught in fall 2009, but not in spring 2010.

So, as opposed to union in the last slide you do a set difference to get this result. So, this is how you can use set theoretic operation to get different relational results this is also the result from the earlier example.

(Refer Slide Time: 11:49)



Set-Intersection Operation

- Notation: $r \cap s$
- Defined as:

$$r \cap s = \{t \mid t \in r \text{ and } t \in s\}$$
- Assume:
 - r, s have the same arity
 - attributes of r and s are compatible
- Note: $r \cap s = r - (r - s)$

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3



s

A	B
α	2

$r \cap s$


SWAYAM: NPTEL-NOC MOOC's Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018
 Database System Concepts - 8th Edition

12.12

Set intersection can be supported is supported, but it is not a basic operation because as it is defined by all tuples which belong to both r and s . It can actually be computed by applying set difference twice and certainly for set intersection also the same assumption about arity and compatibility of types hold and this is the earlier example.

(Refer Slide Time: 12:20)



Cartesian-Product Operation

PPD

- Notation $r \times s$
- Defined as:

$$r \times s = \{t \mid t \in r \text{ and } q \in s\}$$
- Assume that attributes of $r(R)$ and $s(S)$ are disjoint. (That is, $R \cap S = \emptyset$)
- If attributes of $r(R)$ and $s(S)$ are not disjoint, then renaming must be used

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b


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

$r \times s$

SWAYAM: NPTEL-NOC MOOC's Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018
 Database System Concepts - 8th Edition

12.13

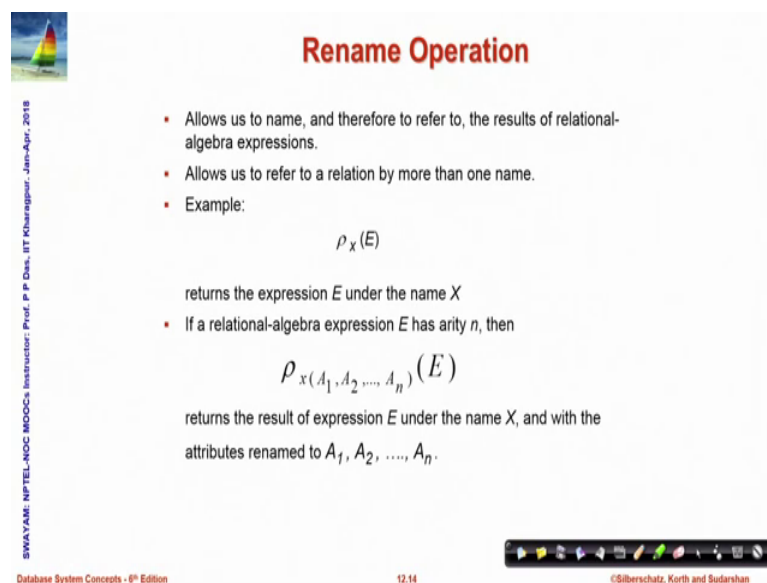


Next is Cartesian product, where we take 2 relations and for the Cartesian product we make we juxtapose 1 relation with the other. So, t is a relation from r q is a tuple from s and we put them side by side to get a tq rho in the Cartesian product r

cross s , which basically means that you compute all possible combinations of pupils from r and of s . It is assumed that the attributes of r and s are disjoint that is a schema of r intersection schema of s is null.

If the attributes are not disjoint then we must use renaming which we will soon see, and here is an example that we had shown earlier of r and s computing r process, Cartesian product is a very useful operation particularly for computing join as we have seen in sql already.

(Refer Slide Time: 13:36)



Rename Operation

- Allows us to name, and therefore to refer to, the results of relational-algebra expressions.
- Allows us to refer to a relation by more than one name.
- Example:

$$\rho_X(E)$$

returns the expression E under the name X

- If a relational-algebra expression E has arity n , then

$$\rho_{X(A_1, A_2, \dots, A_n)}(E)$$

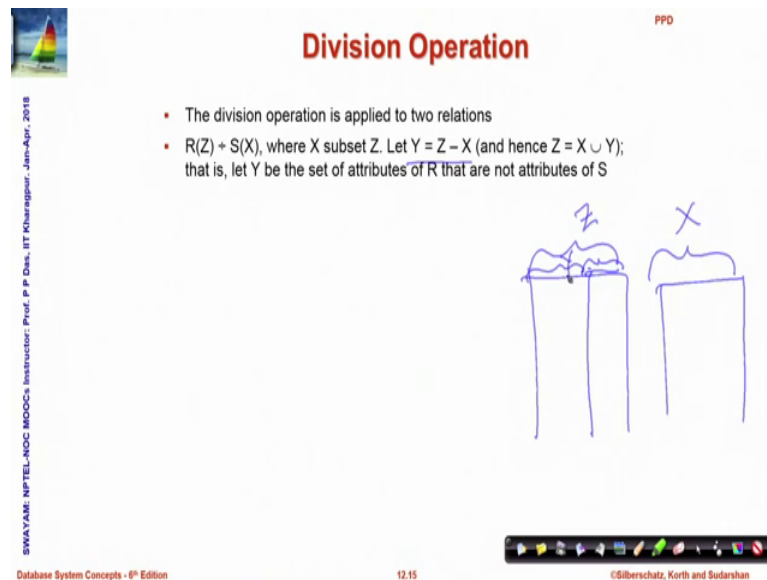
returns the result of expression E under the name X , and with the attributes renamed to A_1, A_2, \dots, A_n .

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.14 ©Silberschatz, Korth and Sudarshan

Rename operation basically allows you to rename some expression attribute into another. So, the operator is rho and you have an expression to which you give the name x and that is how the renaming of you can have multiple attributes of x as well.

(Refer Slide Time: 13:58)



Division Operation

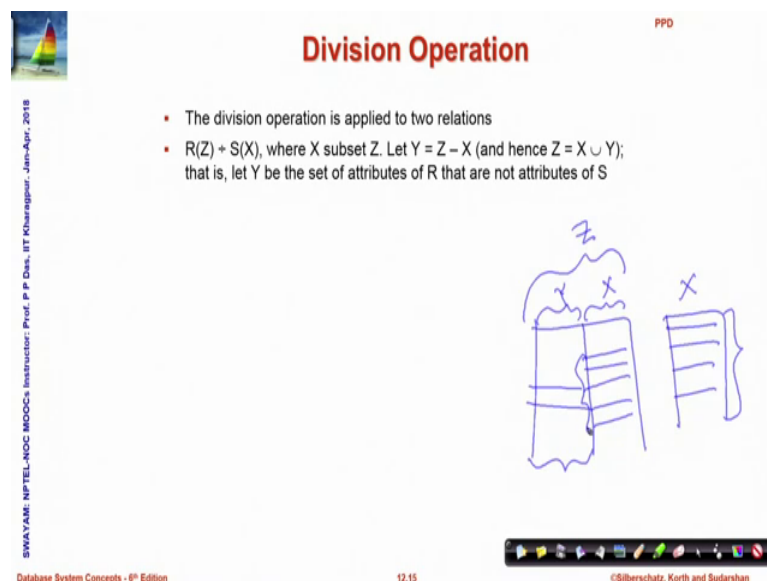
- The division operation is applied to two relations
- $R(Z) \div S(X)$, where $X \subset Z$. Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S

Database System Concepts - 6th Edition 12.15 ©Silberschatz, Korth and Sudarshan

Division is another operator in relational algebra that can be applied between 2 relations, but it is a derived operation. So, it says that if I have, so let me just show you by, by a little bit of sketch that if I have 2 relations which have a set of attributes z and s which is a set of attribute x , such that actually the set z is a superset of x . So, z has more attributes than the relation r has more attributes.

So, if you take the difference of attributes between z and x and call it y then we are interested what happens on these remaining set of attributes y .

(Refer Slide Time: 15:02)



Division Operation

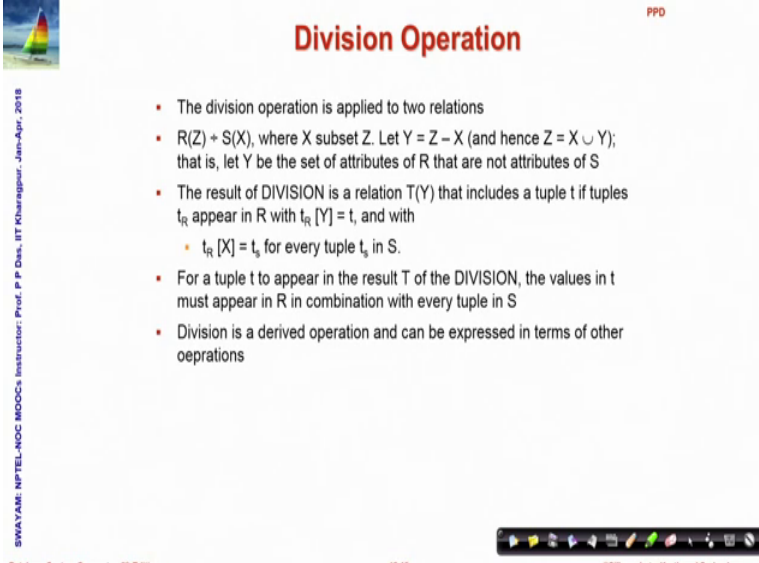
- The division operation is applied to two relations
- $R(Z) \div S(X)$, where $X \subset Z$. Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S

Database System Concepts - 6th Edition 12.15 ©Silberschatz, Korth and Sudarshan

So, this is what, we have this is my x set of attributes this is where x occurs this difference is y this whole set is z . Now in this what we want is we want to in the output we want a relation having only the y attribute such that for every tuple in that relation if I consider all the tuples of s then their cross product must be a part of r .

So, for every tuple here if there are say 4 tuples here, a tuple here must have all these 4 tuples along with it in the result. If it does not have any 1 or more of them then that tuple will not feature in the final result.

(Refer Slide Time: 15:58)



Division Operation

- The division operation is applied to two relations
- $R(Z) \div S(X)$, where X subset Z . Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S
- The result of DIVISION is a relation $T(Y)$ that includes a tuple t if tuples t_R appear in R with $t_R[Y] = t$, and with
 - $t_R[X] = t_s$ for every tuple t_s in S .
- For a tuple t to appear in the result T of the DIVISION, the values in t must appear in R in combination with every tuple in S
- Division is a derived operation and can be expressed in terms of other operations

SWAYAM: NPTEL-NOC MBOCs Instructor: Prof. P. P. Das, IIT Khargpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.15 ©Silberschatz, Korth and Sudarshan

So, the result of a division is a relation TY that include tuple t if tuples tr that is the part of the tuple the tuple that appear in r and on that on the y part the difference part it matches. So, that you have that tr_x is equal to t_s where t_s actually exist in s .

This must happen for all tuples in s , so division is a very good interesting operator which is often required for coding different queries. So, it is a derived operation.

(Refer Slide Time: 16:44)

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

B
1
2

$r \div s$:

A
α
β

e.g.
A is customer name
B is branch-name
1 and 2 here show two specific branch-names
(Find customers who have an account in all branches of the bank)

Source: db.fcgroup.nl/silberslides/Division%20-%20Slides%20-%20relational%20algebra.pptx
Database System Concepts - 6th Edition 12.16 ©Silberschatz, Korth and Sudarshan

Let us take an example, let us say this is the relation r and this is a relation s and I am trying to compute r divided by s .

So, what I want is over the attributes of this is therefore, x this is x this attribute y attribute set y . So, all over x all the values that I have, I must have those values in the relation r , if I do then the attribute the particular tuple matching on the attribute y goes onto the result. So, α goes onto the result because you have both α α 1 as well as α 2 in the set r , in the relation r .

(Refer Slide Time: 17:44)

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

B
1
2

$r \div s$:

A
α
β

e.g.
A is customer name
B is branch-name
1 and 2 here show two specific branch-names
(Find customers who have an account in all branches of the bank)

Source: db.fcgroup.nl/silberslides/Division%20-%20Slides%20-%20relational%20algebra.pptx
Database System Concepts - 6th Edition 12.16 ©Silberschatz, Korth and Sudarshan

Beta 1 is there and beta 2 is also there, so beta also goes into the relation alpha goes in because 1 is there 2 is also there, but gamma will not go in because I have gamma 1, but I do not have a tuple gamma 2 in r if I had gamma 2 in r that will go in the result.

(Refer Slide Time: 18:13)

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

$r \div s$

e.g.
A is customer name
B is branch-name
 1 and 2 here show two specific branch-names
 (Find customers who have an account in all branches of the bank)

Source: db.fcgroup.nl/silberslides/Division%20-%20Slides%20-%20relational%20algebra.pptx
 Database System Concepts - 9th Edition 12.16 ©Silberschatz, Korth and Sudarshan

So, if I can say it again the whole of the relation s must happen over the x attributes of r, consider these 2 together. If that happens then the attributes on y the tuples would be chosen and that is how we get the result having alpha and beta.

(Refer Slide Time: 16:47)

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

$r \div s$

e.g.
Students who have taken both "a" and "b" courses, with instructor 1
 (Find students who have taken all courses given by instructor 1)

Source: db.fcgroup.nl/silberslides/Division%20-%20Slides%20-%20relational%20algebra.pptx
 Database System Concepts - 9th Edition 12.17 ©Silberschatz, Korth and Sudarshan

Let us look at 1 more example, so this is got r has 5 attributes this has x as 2. So, this is this is 2 attributes x, these are 3 attributes y and what I have to look for is those tuples in r where the values over y would be same and I should be able to get the whole table of x over whole table of the relation s over the x attributes. So, if we look at here this is a 1, b 1, a 1, b 1, here these are identical.

So, this particular tuple will go into the result if I look in here this tuple will go into the result, but if I consider this tuple beta a gamma which has a 1 over d, but beta a gamma does not have b 1 it has b 3, so it will not go into the result. So, if you conceptually look at that is the reason this is called a division. So, you get this here you get this here. So, this is like the way we divide that this is the whole and wherever it goes in if the tuples that are identical on the white set of attributes we will collect them into the final result.

(Refer Slide Time: 20:30)

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


e.g.
Students who have taken both "a" and "b" courses, with instructor "1"
(Find students who have taken all courses given by instructor 1)

Source: db.fcgroup.nl/silberslides/Division%20-%20Slides%20-%20relational%20algebra.pptx

Database System Concepts - 9th Edition 12.17 ©Silberschatz, Korth and Sudarshan

So, this is the division operation which by which we can compute the students who have taken both a and b courses instructor 1 will be found out from this division operation.

(Refer Slide Time: 20:51)




Formal Definition

- A basic expression in the relational algebra consists of either one of the following:
 - A relation in the database
 - A constant relation
- 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 consisting of some of the attributes in E_1
 - $\rho_X(E_1)$, X is the new name for the result of E_1


SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.18 ©5th



so formally speaking a basic expression in relational algebra consists either of a relation in the database, which is an instance or a constant relation which does not change which is given. And then we have six operations of union set difference cross product, selection projection, and renaming that can give us all sorts of different relational algebra formula and also the derived operations and whatever we have seen of sql can be expressed in terms of this relational algebra formula.

(Refer Slide Time: 21:30)




PPD

- Relational Algebra
- **Tuple Relational Calculus**
- Domain Relational Calculus
- Equivalence of Algebra and Calculus

TUPLE RELATIONAL CALCULUS


SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.19 ©5th



Now, relational algebra is not something totally unique the same thing can be done in terms of other formulations also.

(Refer Slide Time: 21:41)




Tuple Relational Calculus

- A nonprocedural query language, where each query is of the form $\{t \mid P(t)\}$
- It is the set of all tuples t such that predicate P is true for t
- t is a *tuple variable*, $t[A]$ denotes the value of tuple t on attribute A
- $t \in r$ denotes that tuple t is in relation r
- P is a *formula* similar to that of the predicate calculus

SWAYAM: NPTEL-NOC MBOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018


Database System Concepts - 9th Edition 12.20 ©2018



A second formulation which is also used is known as tuple relational calculus, which is non-procedural relational algebra was procedural because you are actually doing the explaining what the operations or you are detailing out what the operations are in tuple relational calculus you are specify what the condition is you are specifying what this condition is.

So, those tuples which satisfy this condition form the relation, so p is a predicate. So, whatever t satisfies the predicate are included and if a is an attribute then $t[A]$ will denote the value of the tuple on attribute A , A could be a single attribute it could be A set of attributes also and t is a relation that belongs to r , p as I said it is a its a predicate calculus formula.

(Refer Slide Time: 22:43)




Predicate Calculus Formula

1. Set of attributes and constants
2. Set of comparison operators: (e.g., $<$, \leq , $=$, \neq , $>$, \geq)
3. Set of connectives: and (\wedge), or (\vee), not (\neg)
4. Implication (\Rightarrow): $x \Rightarrow y$, if x is true, then y is true
$$x \Rightarrow y \equiv \neg x \vee y$$
5. Set of quantifiers:
 - ▶ $\exists t \in r(Q(t))$ = "there exists" a tuple t in relation r such that predicate $Q(t)$ is true
 - ▶ $\forall t \in r(Q(t))$ = Q is true "for all" tuples t in relation r

SWAYAM: NPTEL-NODD MOOCs Instructor: Prof. P. P. Das, IIT Khargpur, Jan-Apr, 2018

Database System Concepts - 8th Edition

12.21

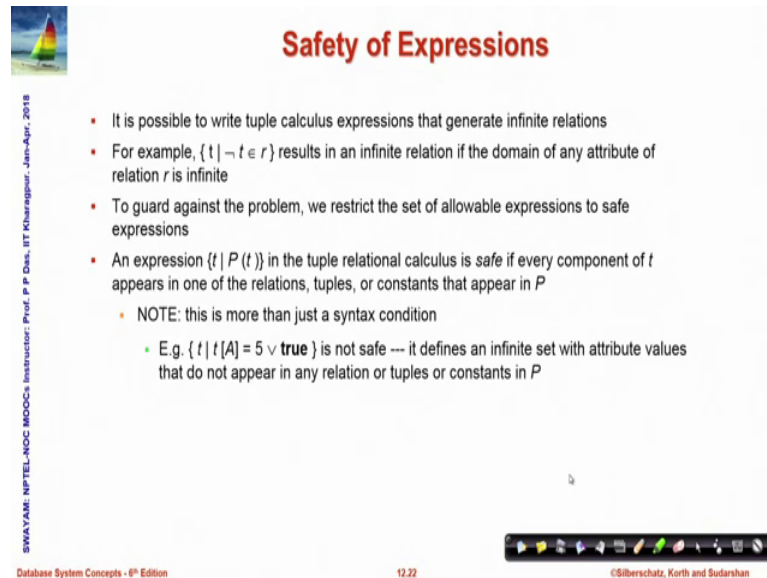


So, it could be a set of attributes or constants this I am just included for your help if in case you have become rusted with predicate calculus you can refer to the a predicate calculus as a set of attributes and constant. It has set of comparison operators the six of them, there are a set of connectives these are all same as the propositional calculus there is implication which says if x is true then y is true if x is false then the whole thing is true vacuously..

And what makes it primarily predicate calculus is a fact that it has existential quantifier, which says that the formula there exists t belongs to r $Q(t)$ holds if I can find at least 1 tuple t which satisfies $Q(t)$.

Similarly, there is a universal quantifier where I will say that for all t belongs to r $Q(t)$ is true if for all tuples of r t satisfies $Q(t)$. So, this in tuple relational calculus all conditions all predicates are formula of this kind and with that we can represent any.

(Refer Slide Time: 24:07)



Safety of Expressions

- It is possible to write tuple calculus expressions that generate infinite relations
- For example, $\{t \mid \neg t \in r\}$ results in an infinite relation if the domain of any attribute of relation r is infinite
- To guard against the problem, we restrict the set of allowable expressions to safe expressions
- An expression $\{t \mid P(t)\}$ in the tuple relational calculus is *safe* if every component of t appears in one of the relations, tuples, or constants that appear in P
 - NOTE: this is more than just a syntax condition
 - E.g. $\{t \mid t[A] = 5 \vee \text{true}\}$ is not safe --- it defines an infinite set with attribute values that do not appear in any relation or tuples or constants in P

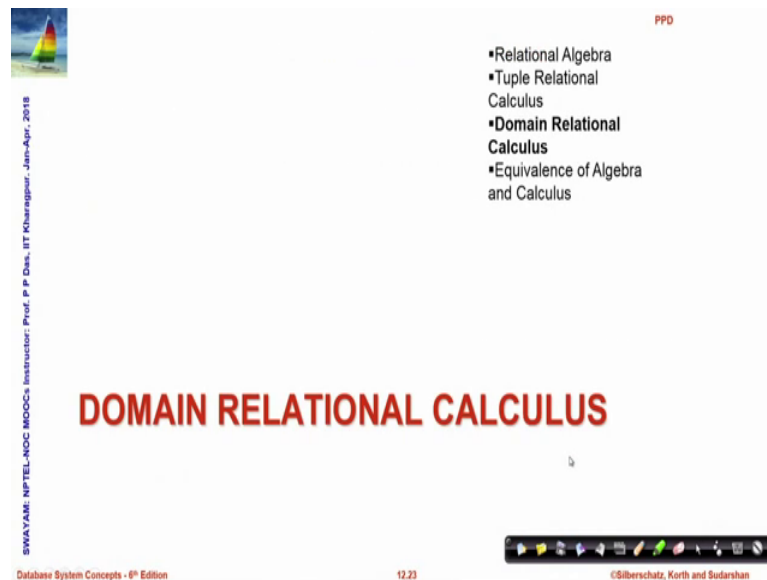
SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Khargpur, Jan-April, 2018

Database System Concepts - 6th Edition 12.22 ©Silberschatz, Korth and Sudarshan

Relational set in full there is a word of caution because it is possible to write tuple relational calculus expression that can potentially generate infinite relations. Now, infinite relations are naturally not representable for example, if I write simply this that r is a relation and I write this predicate that not of t belongs to r , which is basically complement set of r now a complement set of r potentially may be infinite if the domain is infinite..

So, such expressions tuple relational expressions are not acceptable as a part of the design. So, whenever we want to do this we would like to guard this by putting some additional condition and we have to make sure that any expression that we have in tuple relational calculus is a safe expression, in the sense that it does give me finite number of tuples in the relation.

(Refer Slide Time: 25:14)



PPD

- Relational Algebra
- Tuple Relational Calculus
- **Domain Relational Calculus**
- Equivalence of Algebra and Calculus

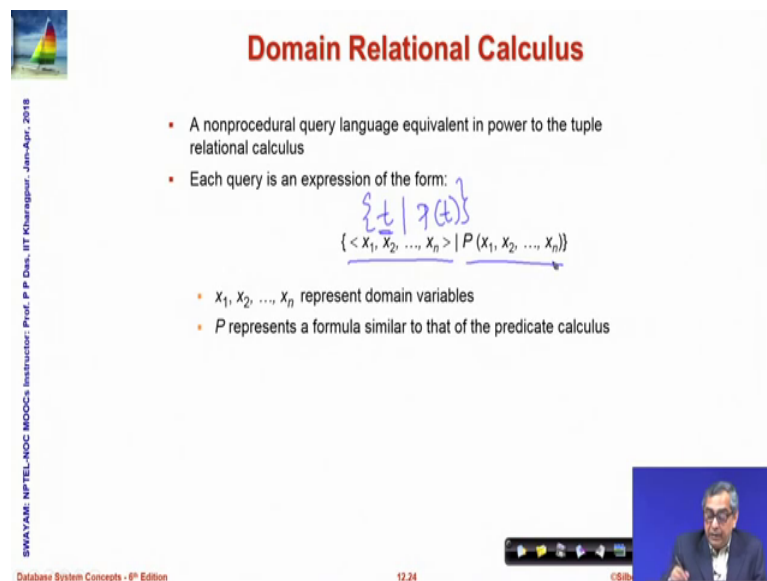
DOMAIN RELATIONAL CALCULUS

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.23 ©Silberschatz, Korth and Sudarshan

A third formalism that exists that is used is known as domain relational calculus.

(Refer Slide Time: 25:25)



Domain Relational Calculus

- A nonprocedural query language equivalent in power to the tuple relational calculus
- Each query is an expression of the form:

$$\{ \langle x_1, x_2, \dots, x_n \rangle \mid P(x_1, x_2, \dots, x_n) \}$$

- x_1, x_2, \dots, x_n represent domain variables
- P represents a formula similar to that of the predicate calculus

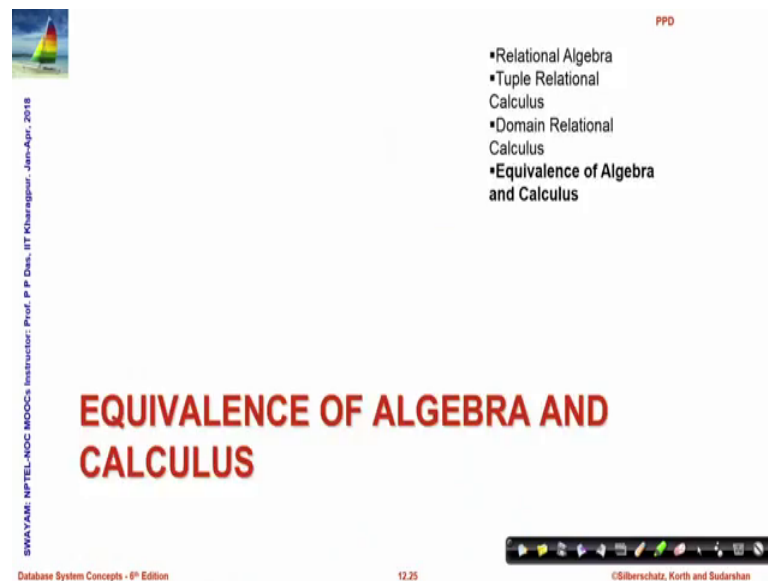
SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.24 ©Silberschatz, Korth and Sudarshan

Which is also non procedural and equivalent in power to tuple relational calculus again, which is very similar to tuple relational calculus the only difference being if you just recall tuple relational calculus. We are writing collection of tuples t such that Pt that is the predicate P is satisfied by t here, instead of writing a tuple variable t we write expand it out in terms of all its components..

So, we write the values of the different components of t over different n attributes and write it as a n tuple and so here instead of having 1 variable t we have n variable x_1 to x_n and therefore, the predicate is formed of this n variables where x_1 to x_n are represent the different domain values, domain variables and that leads to the reason for the name domain relational calculus.

(Refer Slide Time: 26:28)



Now, of the 3 formalisms that we have seen we will not go into direct mathematical proofs, but in the next couple of slides, I just show that they are equivalent in nature. What means that if I can write an expression in relational algebra then it is possible to write an equivalent expression in tuple relational calculus and in domain relational calculus and vice versa..

So, if I can write an expression in any 1 of these formalisms then there are equivalent expressions in the other 2 formalisms as well which is probably very easy to see between, tuple relational calculus and domain relational calculus because 1 is just representing the whole tuple as a single variable whereas, the other is representing it in terms of n domain variables.

So, their equivalence is pretty much very similar the fact that your predicate calculus formula has to change, but it is not, so obvious for the equivalence between relational algebra and the calculi.

(Refer Slide Time: 27:41)

Equivalence of RA, TRC and DRC

Select Operation

$R = (A, B)$

Relational Algebra: $\sigma_{B=17}(r)$

Tuple Calculus: $\{t \mid t \in r \wedge B = 17\}$

Domain Calculus: $\{ \langle a, b \rangle \mid \langle a, b \rangle \in r \wedge b = 17 \}$

Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

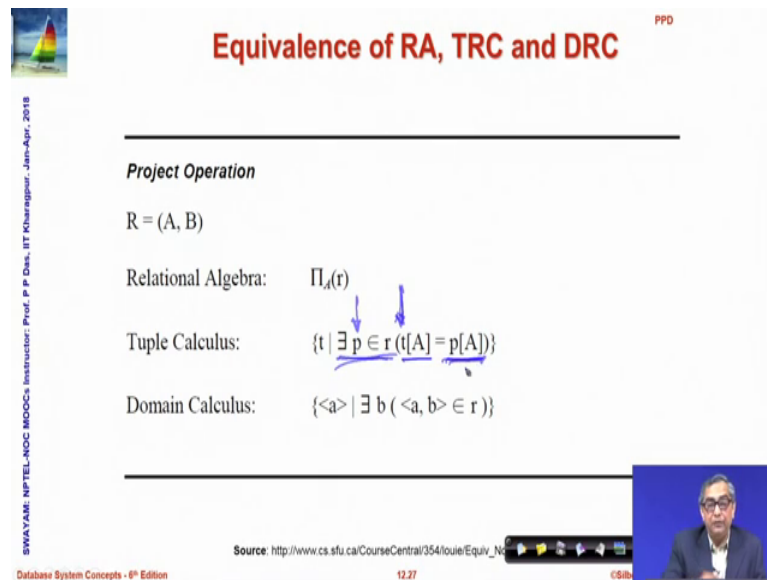
Database System Concepts - 8th Edition 12.26

So, we just show a few examples of the basic operations for example, say select operation. So, I am just not showing the proof, I am just giving you some example cases to show through a relation r has 2 attributes A, B this is what you wanted to write in relational algebra that you want to collect all tuples where B is equal to 17.

Naturally in tuple relational calculus you can easily write the first condition is you are doing it on r . So, t must belong to r and your condition is B should be 17. So, this predicate will represent the same set or the same relation as in tuple calculus in domain calculus there are 2 components a and b . So, you have to say component taken together must belong to r and the component b must be equal to 17.

So, you can see that it is pretty straightforward to see the equivalence between a relational algebra expression involving select and the corresponding tuple calculus or domain calculus expressions this is through an example, but you can certainly easily generalize this as a proof.

(Refer Slide Time: 28:49)



Equivalence of RA, TRC and DRC

Project Operation

$R = (A, B)$

Relational Algebra: $\Pi_A(r)$

Tuple Calculus: $\{t \mid \exists p \in r (t[A] = p[A])\}$

Domain Calculus: $\{ \langle a \rangle \mid \exists b (\langle a, b \rangle \in r) \}$

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. Das, IIT Kharagpur, Jan-April, 2018


Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

Database System Concepts - 8th Edition 12.27

Similarly, for projection if we do a projection on a then all that we are trying to do is we are trying to create a new relation where only the a attribute exists. So, in the tuple t the a attribute exists and if I have projected and got this tuple t then in my original relation r there must be some tuple p such that on the attribute a they match they are same.

So, it is the same thing in relational algebra we said that keep a and erase everything else here we are saying that if we have been able to get a tuple t which has a value t a then there must be a tuple p in r, which has the same value over the same attribute. So, this condition is equivalent representative of the projection, and the same thing can be written in domain calculus you can go through it carefully and convince yourself.

(Refer Slide Time: 29:59)



Equivalence of RA, TRC and DRC

Combining Operations

$R = (A, B)$

Relational Algebra: $\Pi_A(\sigma_{B=17}(r))$

Tuple Calculus: $\{t \mid \exists p \in r (t[A] = p[A] \wedge p[B] = 17)\}$

Domain Calculus: $\{ \langle a \rangle \mid \exists b (\langle a, b \rangle \in r \wedge b = 17) \}$


SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

Database System Concepts - 8th Edition


12.28

PPD



You can combine this as in the relational algebra as well as in tuple calculus. So, here you apply 1 relation 1 operation and then the other 1 selection then projection here you are combining this is part of projection this is also part of projection, but this condition has come from the selection and get a total predicate calculus predicate which will give you the tuple calculus expression for this combined expression of relational algebra , domain calculus will certainly happen in a similar manner.

(Refer Slide Time: 30:34)



Equivalence of RA, TRC and DRC

Union

$R = (A, B, C) \quad S = (A, B, C)$

Relational Algebra: $r \cup s$

Tuple Calculus: $\{t \mid t \in r \vee t \in s\}$

Domain Calculus: $\{ \langle a, b, c \rangle \mid \langle a, b, c \rangle \in r \vee \langle a, b, c \rangle \in s \}$


SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

Database System Concepts - 8th Edition


12.29

PPD



Union certainly straightforward, so you can do it yourself.

(Refer Slide Time: 30:40)



Equivalence of RA, TRC and DRC

Set Difference

$R = (A, B, C) \quad S = (A, B, C)$



Relational Algebra: $r - s$

Tuple Calculus: $\{t \mid t \in r \wedge t \notin s\}$

Domain Calculus: $\{ \langle a, b, c \rangle \mid \langle a, b, c \rangle \in r \wedge \langle a, b, c \rangle \notin s \}$


Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

Database System Concepts - 9th Edition 12.30



Set difference is again very straight forward because that is. In fact, in set theoretically whatever operations we have their relational algebraic definition itself is a tuple calculus formula you can expand them out and write in the domain calculus as well.

(Refer Slide Time: 30:57)



Equivalence of RA, TRC and DRC

Intersection

$R = (A, B, C) \quad S = (A, B, C)$



Relational Algebra: $r \cap s$

Tuple Calculus: $\{t \mid t \in r \wedge t \in s\}$

Domain Calculus: $\{ \langle a, b, c \rangle \mid \langle a, b, c \rangle \in r \wedge \langle a, b, c \rangle \in s \}$


Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

Database System Concepts - 9th Edition 12.31



Intersection plays out in the same way tuples that belong to both r and s .

(Refer Slide Time: 31:04)



Equivalence of RA, TRC and DRC

Cartesian/Cross Product

$R = (A, B) \quad S = (C, D)$


Relational Algebra: $r \times s$

Tuple Calculus: $\{t \mid \exists p \in r \exists q \in s (t[A] = p[A] \wedge t[B] = p[B] \wedge t[C] = q[C] \wedge t[D] = q[D])\}$

Domain Calculus: $\{ \langle a, b, c, d \rangle \mid \langle a, b \rangle \in r \wedge \langle c, d \rangle \in s \}$

Source: http://www.cs.sfu.ca/CourseCentral/354/fouie/Equiv_N

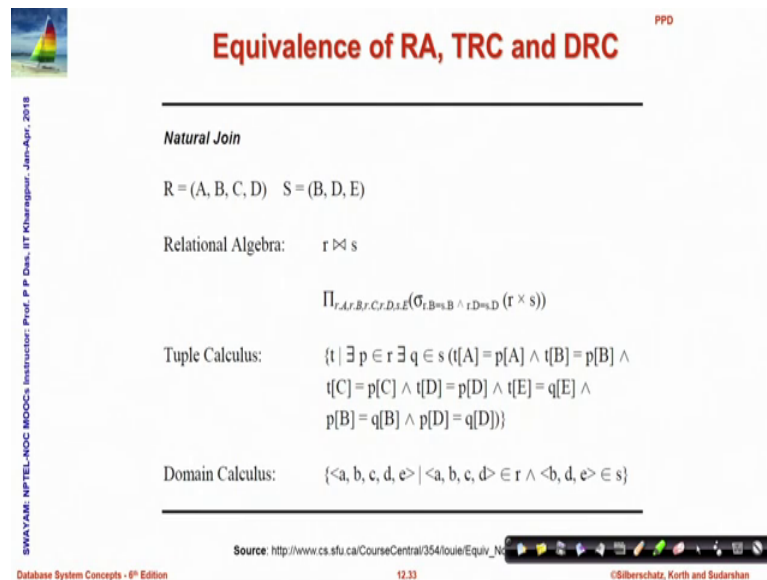
Database System Concepts - 8th Edition 12.32



Cartesian product is a little bit more involved because all that you are saying here is if I have a Cartesian product then if that product has a tuple t . Then there must be a tuple p in the relation r the left relation there must be a tuple q in the in s the right relation. So, that the final tuple t matches p on the a attributes the b attributes that is attributes of relation r and the components of t matches the tuple q in the attributes of s .

If all these conditions happen together then naturally this tuple t is a valid tuple for the Cartesian product. So, you could take examples and work this out and convince yourself that these are really equivalent.

(Refer Slide Time: 31:58)



Equivalence of RA, TRC and DRC

Natural Join

$R = (A, B, C, D) \quad S = (B, D, E)$

Relational Algebra: $r \bowtie s$

$$\Pi_{r.A, r.B, r.C, r.D, s.E}(\sigma_{r.B=s.B \wedge r.D=s.D}(r \times s))$$

Tuple Calculus: $\{t \mid \exists p \in r \exists q \in s (t[A] = p[A] \wedge t[B] = p[B] \wedge t[C] = p[C] \wedge t[D] = p[D] \wedge t[E] = q[E] \wedge p[B] = q[B] \wedge p[D] = q[D])\}$

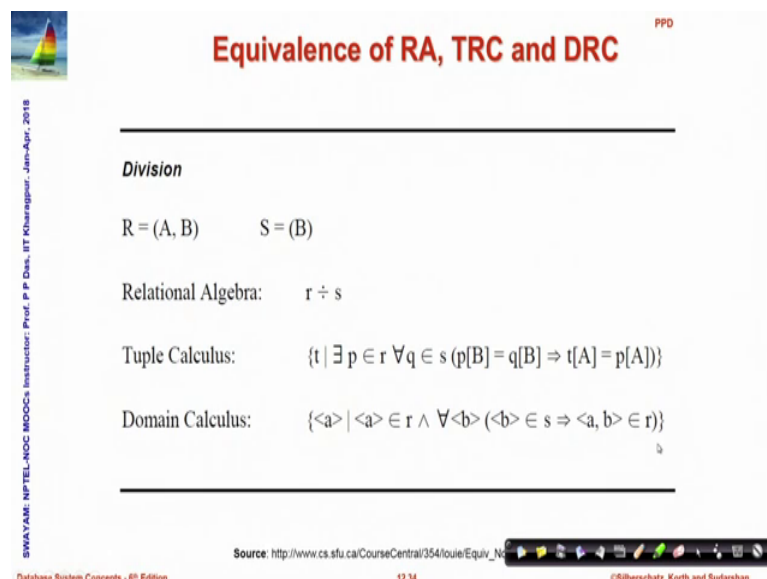
Domain Calculus: $\{ \langle a, b, c, d, e \rangle \mid \langle a, b, c, d \rangle \in r \wedge \langle b, d, e \rangle \in s \}$

Source: http://www.cs.sfu.ca/CourseCentral/354/f04/equiv_NatJoin.html

Database System Concepts - 9th Edition 12.33 ©Silberschatz, Korth and Sudarshan

We can define a natural joint in a similar way, which I will leave it as an exercise for you to convince yourself that this relational algebra expression for natural joint indeed has similar equivalents in tuple and domain calculi.

(Refer Slide Time: 32:17)



Equivalence of RA, TRC and DRC

Division

$R = (A, B) \quad S = (B)$

Relational Algebra: $r \div s$

Tuple Calculus: $\{t \mid \exists p \in r \forall q \in s (p[B] = q[B] \Rightarrow t[A] = p[A])\}$

Domain Calculus: $\{ \langle a \rangle \mid \langle a \rangle \in r \wedge \forall \langle b \rangle (\langle b \rangle \in s \Rightarrow \langle a, b \rangle \in r) \}$

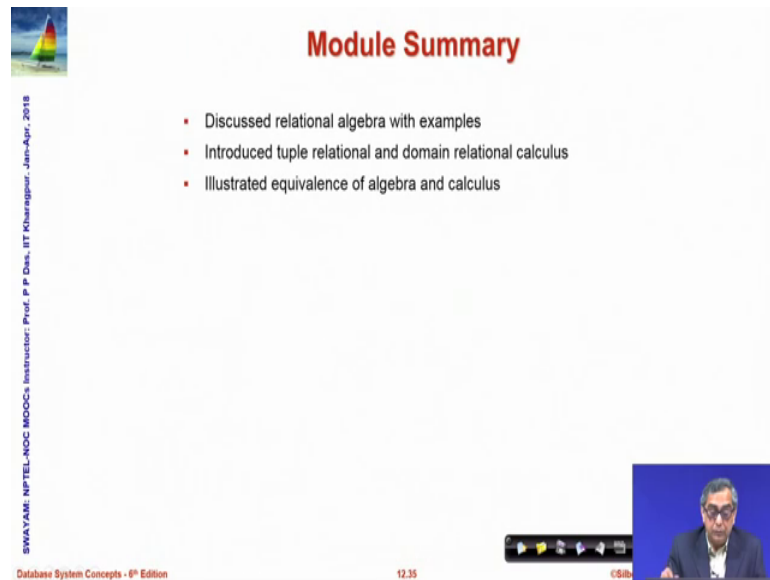
Source: http://www.cs.sfu.ca/CourseCentral/354/f04/equiv_NatJoin.html

Database System Concepts - 9th Edition 12.34 ©Silberschatz, Korth and Sudarshan

Division we just showed as a derived operation, we have not showed how in relational algebra you can write division using the other operations. I will leave that as an exercise as well, but here what I show is in tuple calculus how you can write division using the quantifiers.

Here you can see that here for the first time we do need to use the universal quantifier to make sure that while I divide that the whole of the table of s must be available against the part of the tuple part of the y attributes as we said that will be collected in the result.

(Refer Slide Time: 33:02)



The slide is titled "Module Summary" in red text. It contains a bulleted list of three items: "Discussed relational algebra with examples", "Introduced tuple relational and domain relational calculus", and "Illustrated equivalence of algebra and calculus". On the left side, there is a vertical text string: "SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018". At the bottom left, it says "Database System Concepts - 8th Edition". In the center bottom, the number "12.35" is displayed. On the right side, there is a small video inset showing a man with glasses and a blue background, and a set of navigation icons.

Module Summary

- Discussed relational algebra with examples
- Introduced tuple relational and domain relational calculus
- Illustrated equivalence of algebra and calculus

SWAYAM: NPTEL-NOC MOOCs Instructor: Prof. P. P. Das, IIT Kharagpur, Jan-Apr, 2018

Database System Concepts - 8th Edition 12.35

So, to summarize we have discussed primarily the relational algebra with some examples, we have introduced the tuple relational and domain relational calculus and through a set of examples. We have shown that we have illustrated that the algebra and the calculi are equivalent and I would request you to work out more examples to understand the equivalence, or if you are really enthused please try out proving their equivalence formally as well.