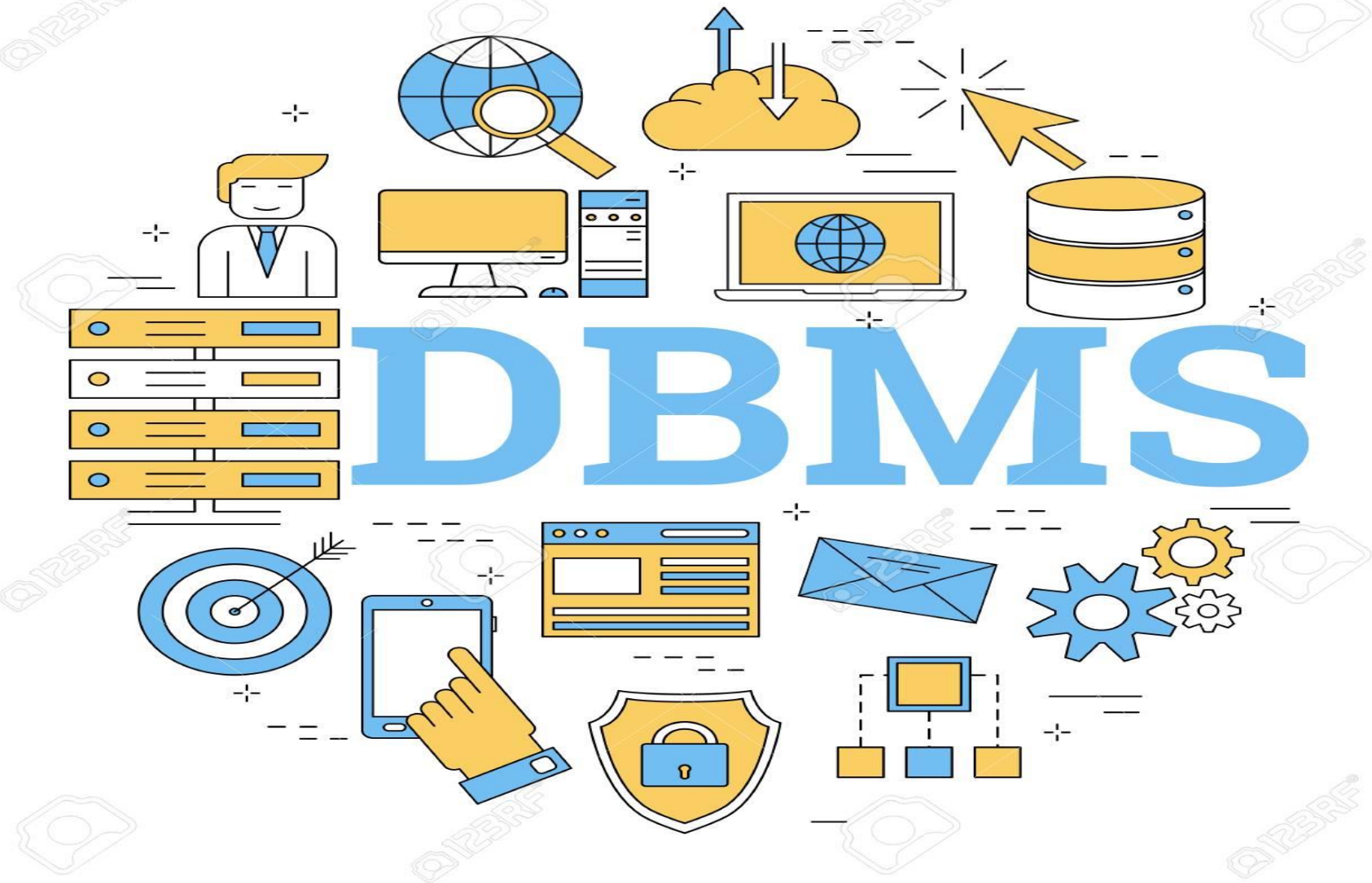
SVKM'S
NMIMS
Deemed to be UNIVERSITY



SVKM'S NMIMS
Deemed to be UNIVERSITY

CHAPTER 4
Relational Algebra

- Introduction to Relational Algebra
- Set Operations: Union , Join and Difference
- Multi-table Queries: Cartesian Product , Joins , Subqueries
- Extended Operations- Extended Projection , Division
- Practice Questions

SVKM'S NMIMS
Deemed to be UNIVERSITY

WHY RELATIONAL ALGEBRA ??

- In SQL we write WHAT we want to get form the data
- The database system needs to figure out HOW to get the data we want
- The passage from WHAT to HOW goes through the Relational Algebra

SQL = WHAT

Write the names of those customer who lives in Seattle and have purchased a product of price greater than $100.

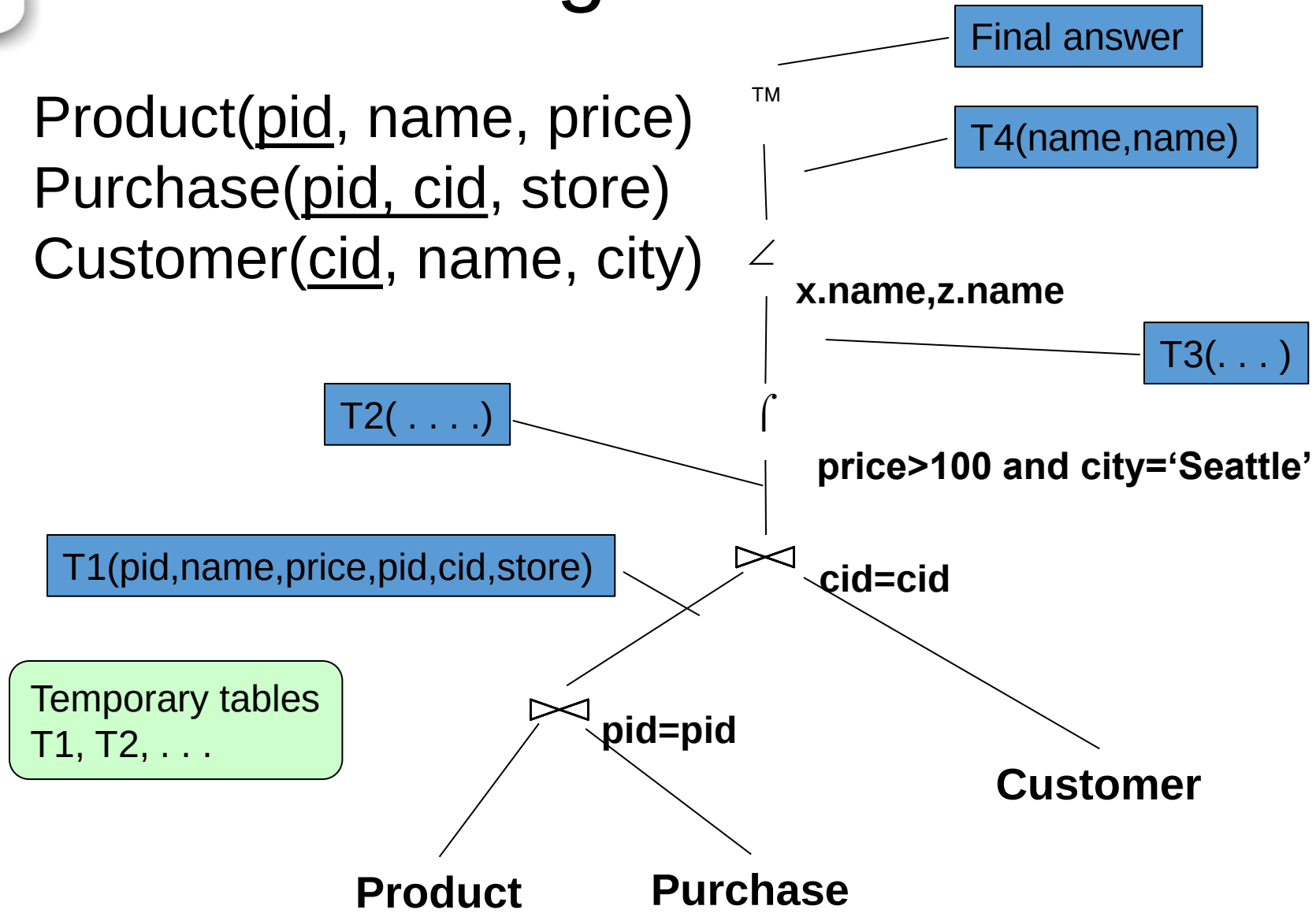
Product(pid, name, price)
Purchase(pid, cid, store)
Customer(cid, name, city)

```
SELECT DISTINCT x.name, z.name
FROM Product x, Purchase y, Customer z
WHERE x.pid = y.pid and y.cid = z.cid and
      x.price > 100 and z.city = 'Seattle'
```

It's clear WHAT we want, unclear HOW to get it

Relational Algebra = HOW

Product(pid, name, price)
Purchase(pid, cid, store)
Customer(cid, name, city)



Relational Algebra

- It is procedural query language.
- It consists of set of operations that take one or two relations as input and produce a new relation as their result.
- The fundamental relational algebra are:
  - select: σ
  - project: Π
  - union: $\cup$
  - set difference: –
  - Cartesian product: x
  - rename: ρ

- **Fundamental Operations**

1. **SELECT**

- It is denoted by sigma (**σ**)
- The select operation selects tuples that satisfy the given predicate.
- The predicate appears as a subscript to **σ.**
- The augment relation is in parenthesis after as the σ

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

(a) The *instructor* table

- $\sigma_{dept\_name = \text{“Physics”}}$ **(instructor)**
- $\sigma_{salary > 90000}$**(instructor)**
- $\sigma_{dept\_name = \text{“Physics”} \wedge salary > 90000}$ **(instructor)**

2. PROJECT operation

- Suppose we want to list all instructors ID, name and salary but not dept_name.
- The project operation allows us to produce this relation.
- The project operation is unary operation.
- It eliminates duplicates rows.
- It is denoted by pi (**Π).**

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

(a) The *instructor* table

- $\Pi_{ID,NAME,DEPT\_NAME}$**(INSTRUCTOR)**

ID	*name*	*dept_name*
22222	Einstein	Physics
12121	Wu	Finance
32343	El Said	History
45565	Katz	Comp. Sci.
98345	Kim	Elec. Eng.
76766	Crick	Biology
10101	Srinivasan	Comp. Sci.
58583	Califieri	History
83821	Brandt	Comp. Sci.
15151	Mozart	Music
33456	Gold	Physics
76543	Singh	Finance

- **Composition of relational operations:**
  - For complicated queries like “Find the name of all instructors in the physics department”.
  - **Π_{name} ($\sigma_{dept\_name = \text{“Physics”}}$ (instructor))**

  - Relational operations can be composed together into a relational algebra expression.

3. **UNION Operation**

Relations r:

A	B
1	α
1	β
2	α

s:

A	B
2	β
2	α

r ∪ s:

A	B
1	α
1	β
2	β
2	α

- 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.

    1. r, s must have the same arity (same number of attributes)

    2. The attribute domains must be compatible (example: 2 nd column of r deals with the same type of values as does the 2 nd column of s)

customer_name	*account_number*
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

DEPOSITOR

customer_name	*loan_number*
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

BORROWER

- Example: To find all customers with either an account or a loan

  ∏customer_name (depositor) ∪ ∏customer_name (borrower)

4. SET DIFFERENCE operation

Relations r:

A	B
1	α
1	β
2	α

s:

A	B
3	β
2	α

r - s:

A	B
1	α
1	β

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

5. Cartesian Product Operation

Relations r :

A	B
α	1
β	2

s:

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

r X s :

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

- Notation r x s
- Defined as: r x s = {t q | t ∈ r and q ∈ s}

Composition of Operations

- Can build expressions using multiple operations

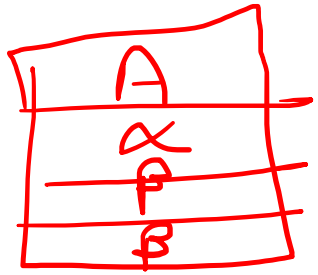
Example:

- r x s

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

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

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



6. 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:

$$\rho_{y}(E)$$

returns the expression E under the name y

ρ_{demo} (instructor)

[from student s

- Example:

Find the highest salary in the university.

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

(a) The *instructor* table

Our Strategy

1) Compute first a temporary relation consisting of those salaries that are not largest
2) Take the set difference between the relation $\Pi_{salary}(instructor)$ and the temporary relation just created.

Step 1: $\Pi_{salary}(\sigma_{\textbf{instructor.salary < d.salary}} (instructor \times \rho_{d\ (instructor)}))$

Step 2: $\Pi_{salary}(instructor) - \Pi_{salary}(\sigma_{\textbf{instructor.salary < d.salary}} (instructor \times \rho_{d\ (instructor)}))$

95000

SVKM'S NMIMS
Deemed to be UNIVERSITY

Practice Questions

Consider Employee(E_no, E_name, dept, salary, age)

Q) Select all employees having salary above 5000 and age above 65

Q) Find salary of all employees having age less than 25.

1. $\Pi_{e\_name}(\sigma_{sal>5000 \wedge age>65}(Employee))$

2. $\Pi_{sal}(\sigma_{age<25}(Employee))$

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)

- Find all loans of over $7600
- Find the loan number for each loan of an amount greater than $7600
- Find the names of all customers who have a loan, an account, or both, from the bank
- Find the names of all customers who have a loan at the Perryridge branch.
- Find the largest account balance

Additional Relational Algebra Operation:

1) Assignment Operation

- 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 always be made to a temporary relation variable.

2) JOIN OPERATION

Join is a combination of a Cartesian product followed by a selection process. A Join operation pairs two tuples from different relations, if and only if a given join condition is satisfied.

Types:

Inner Join: only those tuples that satisfy the matching criteria are included, while the rest are excluded.

(i) Theta Join

(ii) Equi Join

(iii) Natural Join

Outer Join: along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria

(i) Left Outer Join

(ii) Right Outer Join

(iii) Full Outer Join

i) Theta Join

Theta Join allows you to merge two tables based on the condition represented by theta. Theta joins work for all comparison operators.

The general case of JOIN operation is called a Theta join. It is denoted by symbol **θ**

Syntax:

$A \bowtie_{\theta} B$

Theta join can use any conditions in the selection criteria.

Consider the following tables:

Table A			Table B	
column 1	**column 2**		**column 1**	**column 2**
1	1		1	1
1	2		1	3

$A \bowtie_{A.column\ 2 > B.column\ 2} (B)$

ii) EQUI JOIN

When a theta join uses only equivalence condition, it becomes an equi join.

$A \bowtie_{A.column\ 2 = B.column\ 2} (B)$

iii) Natural Join Operation

- Natural join does not utilize any of the comparison operators. In this type of join, the attributes should have the same name and domain. In this type of join, there should be at least one common attribute between two relations.
- It performs selection forming equality on those attributes which appear in both relations and eliminates the duplicate attributes.
- Example:

  $R = (A, B, C, D)$

  $S = (E, B, D)$

  - Result schema = (A, B, C, D, E)
  - $r \bowtie s$ is defined as:

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

- Relation r :

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

s:

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

- 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

DEPT_ID	D_NAME	LOCATION
D1	CSE	A-WING
D2	MECH	B-WING
D3	EXTC	A-WING
D4	CIVIL	B-WING

STUDENT_ID	S_NAME	DEPT_ID
A001	AKSHAR	D1
A002	PAYAL	D1
B001	MAHESH	D2
C004	DIPTI	D3

STUDENT_ID	S_NAME	DEPT_ID	D_NAME	LOCATION
A001	AKSHAR	D1	CSE	A-WING
A002	PAYAL	D1	CSE	A-WING
B001	MAHESH	D2	MECH	B-WING
C004	DIPTI	D3	EXTC	A-WING

- **Outer Joins**

Theta Join, Equijoin, and Natural Join are called inner joins. An inner join includes only those tuples with matching attributes and the rest are discarded in the resulting relation. Therefore, we need to use outer joins to include all the tuples from the participating relations in the resulting relation. There are three kinds of outer joins −

Left outer join

Right outer join

Full outer join

LEFT OUTER JOIN

DEPT_ID	D_NAME	LOCATION
D1	CSE	A-WING
D2	MECH	B-WING
D3	EXTC	A-WING
D4	CIVIL	B-WING

STUDENT_ID	S_NAME	DEPT_ID
A001	AKSHAR	D1
A002	PAYAL	D1
B001	MAHESH	D2
C004	DIPTI	D3

STUDENT_ID	S_NAME	DEPT_ID	D_NAME	LOCATION
A001	AKSHAR	D1	CSE	A-WING
A002	PAYAL	D1	CSE	A-WING
B001	MAHESH	D2	MECH	B-WING
C004	DIPTI	D3	EXTC	A-WING
NULL	NULL	D4	CIVIL	B-WING

RIGHT OUTER JOIN

DEPT_ID	D_NAME	LOCATION
D1	CSE	A-WING
D2	MECH	B-WING
D3	EXTC	A-WING

STUDENT_ID	S_NAME	DEPT_ID
A001	AKSHAR	D1
A002	PAYAL	D1
B001	MAHESH	D2
C004	DIPTI	D5

STUDENT_ID	S_NAME	DEPT_ID	D_NAME	LOCATION
A001	AKSHAR	D1	CSE	A-WING
A002	PAYAL	D1	CSE	A-WING
B001	MAHESH	D2	MECH	B-WING
C004	DIPTI	D5	NULL	NULL

All rows from Left Table.

All rows from Right Table.

FULL OUTER JOIN

DEPT_ID	D_NAME	LOCATION
D1	CSE	A-WING
D2	MECH	B-WING
D3	EXTC	A-WING
D4	CIVIL	B-WING

STUDENT_ID	S_NAME	DEPT_ID
A001	AKSHAR	D1
A002	PAYAL	D1
B001	MAHESH	D2
C004	DIPTI	D5

STUDENT_ID	S_NAME	DEPT_ID	D_NAME	LOCATION
A001	AKSHAR	D1	CSE	A-WING
A002	PAYAL	D1	CSE	A-WING
B001	MAHESH	D2	MECH	B-WING
C004	DIPTI	D5	NULL	NULL
NULL	NULL	D4	CIVIL	B-WING

3. DIVISION OPERATION:

Relation r:

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

s:

B
1
2

r ÷ s

A
α
β

Another Example of Division:

- Relation r:

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

s:

D	E
a	1
b	1

- r ÷ s

A	B	C
α	a	γ
γ	a	γ

User

Id	Name	Age	Gender	OccupationId	CityId
1	John	25	Male	1	3
2	Sara	20	Female	3	4
3	Victor	31	Male	2	5
4	Jane	27	Female	1	3

Occupation

OccupationId	OccupationName
1	Software Engineer
2	Accountant
3	Pharmacist
4	Library Assistant

City

CityId	CityName
1	Halifax
2	Calgary
3	Boston
4	New York
5	Toronto

Q) Select the users whose age is less than 25
Q) Select the users whose id is greater than 2 or age is not 31.
Q) Select the person who stays in Boston.
Q) Select the occupation of every person.

Generalized Projection:

- The generalized projection operation extends the projection operation by allowing arithmetic functions to be used in the projection list.

Customer_name	Limit	Credit_balance
KEN	2000	1750
JONES	6000	700
SMITH	2000	400
HAYES	1500	1500

- If we want to find out how much each person can spend,

Π**customer_name,(limit-credit_balance) as credit_available (CUSTOMER)**

- **Aggregate Functions**

a) SUM
b) COUNT
c) DISTINCT
d) MAX
e) MIN

G SUM (SALARY) (INSTRUCTOR)

Q) Consider the following relations for database that keeps track of student enrolment in courses and books issued for each course.

STUDENT(Ssn, Name, Subject, DOB)

COURSE(Course_id, CName, Dept)

ENROLL(Ssn, Course_id, Semester, Grade)

Book_issued(Course_id, Semester, ISBN)

TEXT(ISBN, Title,Publisher,Author)

1. Write a query to select all courses available in institute.
2. Find all the student details registered for course_id 10.
3. Find various book titles and authors for semester higher than 3.
4. Find all students belong to IT department
5. Find total number of students enrolled in IT Department.

RELATIONAL SCHEMA:

Airport (Airport_ID, Name, city)

Flight(flight_no, flightcompany, depAirport, arrairport)

Booking(ticke_no, name, nationality, flight_no, seatno)

Seat(seat_no, flight_no, class)

Solve the following Questions using relational algebra:

(a) Retrieve all information about airports in London. The schema of the output table should be same as that of the Airport table.

(b) Retrieve details of all bookings by British and French passengers. The schema of the output table should be same as that of the Booking table.

(c) Retrieve the names of all passengers.

(d) Retrieve the flight number, Departure and Arrival airports of all British Airways flights.

(e) Retrieve the name of every passenger together with their flight number and the associated flight company.

(f) Find out the ticket numbers and names of all passengers departing from London.

Pubs Database Schema

author(*author_id*, *first_name*, *last_name*)

author_pub(*author_id*, *pub_id*, *author_position*)

book(*book_id*, *book_title*, *month*, *year*, *editor*)

pub(*pub_id*, *title*, *book_id*)

- *author_id* in *author_pub* is a foreign key referencing *author*
- *pub_id* in *author_pub* is a foreign key referencing *pub*
- *book_id* in *pub* is a foreign key referencing *book*
- *editor* in *book* is a foreign key referencing *author*(*author_id*)
- Primary keys are underlined

Pubs Database State

r(*author*)

author_id	first_name	last_name
1	John	McCarthy
2	Dennis	Ritchie
3	Ken	Thompson
4	Claude	Shannon
5	Alan	Turing
6	Alonzo	Church
7	Perry	White
8	Moshe	Vardi
9	Roy	Batty

r(*author_pub*)

author_id	pub_id	author_position
1	1	1
2	2	1
3	2	2
4	3	1
5	4	1
5	5	1
6	6	1

r(*book*)

book_id	book_title	month	year	editor
1	CACM	April	1960	8
2	CACM	July	1974	8
3	BST	July	1948	2
4	LMS	November	1936	7
5	Mind	October	1950	NULL
6	AMS	Month	1941	NULL
7	AAAI	July	2012	9
8	NIPS	July	2012	9

r(*pub*)

pub_id	title	book_id
1	LISP	1
2	Unix	2
3	Info Theory	3
4	Turing Machines	4
5	Turing Test	5
6	Lambda Calculus	6

Solve the following Questions using relational algebra:

Q) Select book titles.

Q) How many authors are not book editors

Q) Write a relational algebra expression that returns the names of all authors who are book editors

Q) Write a relational algebra expression that returns the names of all authors who have at least one publication in the database.

Q) How many tuples are returned by the following relational algebra expression?

author $\bowtie$ author id=editor book

SVKM'S NMIMS Deemed to be UNIVERSITY

RELATIONAL SCHEMA:

BOOKS(DocId, Title, Publisher, Year)

STUDENTS(StId, StName, Major, Age)

AUTHORS(AName, Address)

borrows(DocId, StId, Date)

has-written(DocId, AName)

describes(DocId, Keyword)

Solve the following Questions using relational algebra:

Q) List the year and title of each book

Q) List all information about students whose major is in CS

Q) List all students with books they can borrow

Q) List all books published by McGraw-Hill before 1990

Q) List the name of those authors who are living in Davis

Q) List the name of students who are older than 30 and who are not studying CS

Q) Rename AName in the relation AUTHORS to Name

Q) List the names of all students who have borrowed a book and who are CS majors.

Q) List each book with its keywords.

Q) List the authors of the books the student 'Smith' has borrowed.

Q) Which books have both keywords 'database' and 'programming'?

SVKM'S NMIMS Deemed to be UNIVERSITY

SCHEMA:

- Actor (actorId, name, nationality, age)
- Film (filmId, title, year, directorId)
- Performance (actorId, filmId, character)
- Director (directorId, name, nationality)

Solve the following Questions using relational algebra:

- Retrieve details of all films that were released in 2010. The output schema should be same as that of the Film table.
- Retrieve details of all actors that are not in their thirties. The output schema should be the same as that of actor table.
- Retrieve the names of all directors.
- Retrieve the names of all American directors.
- Find out the names of all British actors above the age of 40.
- Retrieve the name of each actor together with the titles of the film he/she has performed in.
- Find out the names of all actors that have played the character of Bruce Wayne, together with the year corresponding films were released.
- Retrieve all actors from the film inception.

SVKM'S NMIMS Deemed to be UNIVERSITY

Tuple Relational Calculus

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

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 (v), not ($\neg$)
4. Implication ($\Rightarrow$): x $\Rightarrow$ y, if x if true, then y is true
5. Set of quantifiers:
   - $\exists\, t \in r\,(Q\,(t\,)) \equiv$ "there exists" a tuple in t in relation r such that predicate $Q\,(t\,)$ is true
   - $\forall t \in r\,(Q\,(t\,)) \equiv Q$ is true "for all" tuples t in relation r

Example Queries

RA: $\sigma_{sal>80000}(instructor$

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000

TRC : $R=\{t \mid t \in instructor \wedge t[salary] > 80000\}$

variables

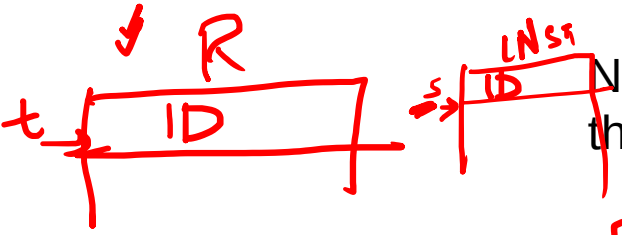
Free

Bounded

↳ Quantifier

Notice that a relation on schema (*ID, name, dept_name, salary*) is implicitly defined by the query

- As in the previous query, but output only the *ID* attribute value



Notice that a relation on schema (*ID*) is implicitly defined by the query

RA : $\pi_{ID}(\sigma_{sal>80000}(instructor))$

TRC

FV

BV

R: $\{t \mid \exists s \in instructor (t[ID] = s[ID] \wedge s[sal] > 80000)\}$

OR $t[ID] = s[ID]$

$s[ID] = t[ID]$

Example Queries

- Find the names of all instructors whose department is in the Watson building

inst

ID	name	dept_name	sal

$R: \{t \mid \exists s \in instructor\ (t[name] = s[name]$

$\wedge \exists u \in dept\ (u[dept\_name] = s[dept\_name] \wedge u[building] = 'Watson'))\}$

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both

section

course_id	sem	year

$\{t \mid \exists s \in section\ (t[course\_id] = s[course\_id] \wedge s[sem] = 'Fall' \wedge s[year] = 2009$

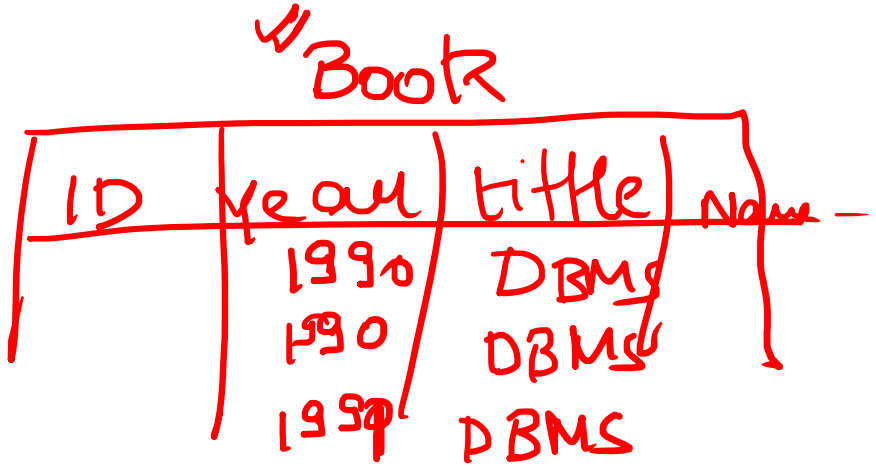
$\wedge \neg \exists u \in section\ (u[course\_id] = t[course\_id] \wedge u[sem] = 'Spring' \wedge u[year]$

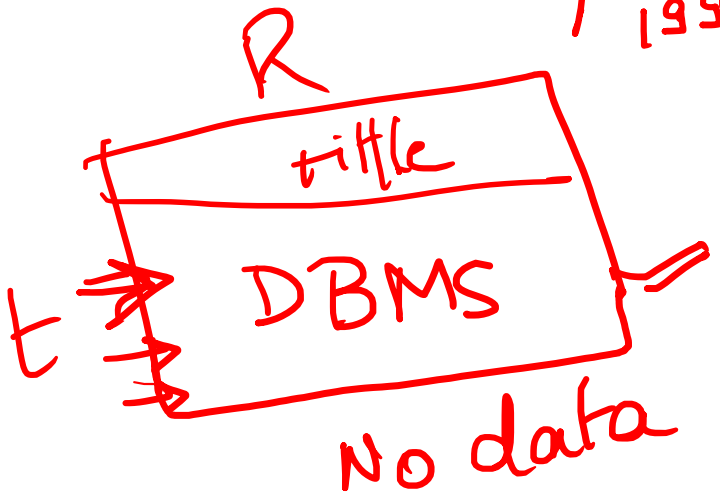
Example Queries

- Find the set of all courses taught in the Fall 2009 semester, and in the Spring 2010 semester

- Find the set of all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

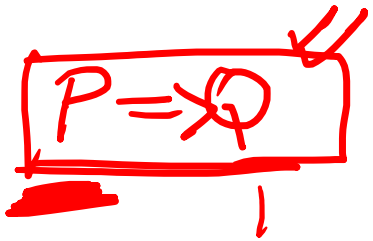
Universal Quantification





$R = \{t \mid \forall u \in book(t[title] = u[title]$

$= u[title]$



$\simeq \neg P \vee Q$

$R = \{t \mid \forall u \in book(t[title] = u[title]$

$\Rightarrow u[book] = 1990$

$P \Rightarrow Q$

SVKM'S NMIMS
Deemed to be UNIVERSITY

Domain Relational Calculus

Domain Relational Calculus

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

$$\{ < x_1, x_2, \ldots, x_n > \mid P\ (x_1, x_2, \ldots, x_n)\}$$

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

Example Queries

- Find the *ID, name, dept_name, salary* for instructors whose salary is greater than $80,000
  - $\{< i, n, d, s> \mid < i, n, d, s> \in instructor \wedge s > 80000\}$
- As in the previous query, but output only the *ID* attribute value
  - $\{< i> \mid < i, n, d, s> \in instructor \wedge s > 80000\}$
- Find the names of all instructors whose department is in the Watson building

$$\{< n > \mid \exists\, i, d, s\, (< i, n, d, s > \in instructor \wedge \exists\, b, a\, (< d, b, a> \in department \wedge b = \text{“Watson”}))\}$$

Example Queries

- Find the set of all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or both

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, r, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009) \vee \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, r, t> \in section] \wedge s = \text{“Spring”} \wedge y = 2010)\}$$

This case can also be written as

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, r, t> \in section \wedge ((s = \text{“Fall”} \wedge y = 2009) \vee (s = \text{“Spring”} \wedge y = 2010))\}$$

- Find the set of all courses taught in the Fall 2009 semester, and in the Spring 2010 semester

$$\{<c> \mid \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, r, t> \in section \wedge s = \text{“Fall”} \wedge y = 2009) \wedge \exists\, a, s, y, b, r, t\ (<c, a, s, y, b, r, t> \in section] \wedge s = \text{“Spring”} \wedge y = 2010)\}$$