CSE 301 – DATABASE

Lecture 7

Chapter 3: Relational Model

Relational Algebra Operations

- Select Operation (σ)
- Project Operation (∏)
- Union Operation (∪)
- Set Difference (-)
- Cartesian Product (X)
- Rename Operation (ρ)

✓ Select, project and rename are unary operator because they operate on one relation.

✓ Union, set difference, cartesian product are binary operator because they operate on two relation.

- Natural Join (⋈)
- Left Outer Join (⟕)
- Right Outer Join (⟖)
- Full Outer Join (⟗)
- Set Intersection (∩)
- Division (÷)
- Assignment (⟵)

❖ All are binary operator because they operate on two relation.

Select Operation (σ)

- Select Operation (σ) is a unary operator in relational algebra that performs a selection operation.
- It select tuples or rows that satisfy the given condition or predicate from a relation.
- It is denoted by sigma (σ)
- Notation: $\sigma_p(r)$ or $\sigma_{(Condition)}(Relation\ Name)$
- P is used as a prepositional logic formula
  - ✓ which may use logical connectives: ∧ (AND), ∨ (OR), ! (NOT) and relational operator like =, ≠, <, >, ≤, ≥ to form the condition
  - ✓ where we specify the conditions that must be satisfied by the data.

Select Operation (σ)

- The WHERE clause of SQL command correspondence to relational select σ()
- SQL: SELECT * FROM R WHERE C;
- E.g. *Select tuples from student table whose age is greater than 17*

$$\sigma_{age>17}(\text{Student})$$

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An

Query 1

Select the student whose roll number is 4.

$$\sigma_{roll\_no\ =\ 4}(Student)$$

Roll_No	Name	Age	Address
4	D	19	Dai An

- ✓ In selection operation, Schema in resulting relation is identical to schema of input relation

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An

Query 2

Select the student whose name is D.

$$\sigma_{name = \text{“D”}} (Student)$$

Roll_No	Name	Age	Address
4	D	19	Dai An

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An

Query 3

Select the student whose age is greater than 17.

$$\sigma_{age>17}(Student)$$

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
4	D	19	Dai An
5	E	18	Dai An

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An

Query 4

Select the student whose age is greater than 17 and who lives in Dai An. .

$$\sigma_{\text{age>17} \wedge \text{address = “Dai An”}}(\text{Student})$$

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
4	D	19	Dai An
5	E	18	Dai An

Examples

- Select tuples from a relation “Books” where subject is “Database”.

$$\sigma_{\text{Subject = “Database”}}(\text{Books})$$

- Select tuples from a relation “Books” where subject is “Database” and price is 450.

$$\sigma_{\text{Subject = “Database”} \wedge \text{price} = 450}(\text{Books})$$

- Select tuples from a relation “Books” where subject is “Database” and price is 450 or have a publication year after 2015.

$$\sigma_{\text{Subject = “Database”} \wedge \text{price} = 450 \vee \text{year} > 2015}(\text{Books})$$

Projection Operator (∏)

- Projection operation is a unary operator in relational algebra that performs a projection operation.
- It projects or display the particular columns or attributes from a relation and
- It delete columns that are not projection list.
- It is denoted by ∏.
- ***Notation***: $\prod_{A1, A2, \ldots, An}$ (r) **or** $\prod_{Attribute\_list}$ (Relation name)
  - Where A1, A2, …, An are the attribute names of relation.

Projection Operator (∏)

- Duplicate rows are automatically eliminated from result.
- The SQL SELECT command corresponds to relational project ∏( ).
- SQL: SELECT A1, A2, …, FROM R;

- **Example:** Display the columns roll_no from the relation Student.

$$\prod_{roll\_no}(\text{Student})$$

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An
6	F	18	Da Lat

Name
A
B
C
D
E
F

Query 1

Display (or project) the name of the students in student table (or relation)

$$\Pi_{Name}(\text{Student})$$

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An
6	F	18	Da Lat

Roll_No	Name
1	A
2	B
3	C
4	D
5	E
6	F

$$\Pi_{\mathit{roll\_no,\ name}}(\text{Student})$$

Query 2

Display (or project) the roll number and name of the students in student table (or relation)

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An
6	F	18	Da Lat

Age
20
17
16
19
18

Duplicate rows are automatically eliminated from result

$$\Pi_{age}\,(\text{Student})$$

Query 3

Display (or project) the age of the students in student table (or relation)

Example

Student

Roll_No	Name	Age	Address
1	A	20	Ho Chi Minh
2	B	17	Hanoi
3	C	16	Hanoi
4	D	19	Dai An
5	E	18	Dai An
6	F	18	Da Lat

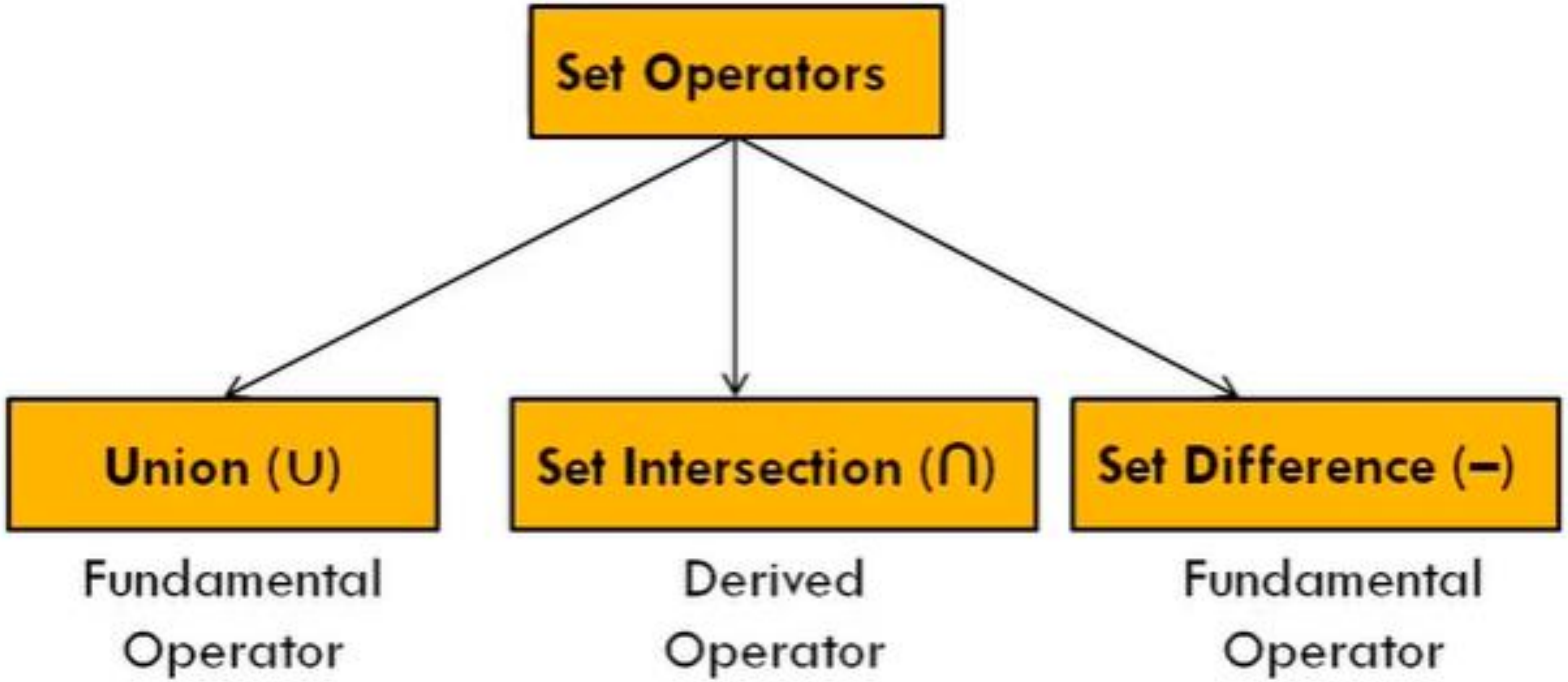
Roll_No	Name
1	A
4	D
5	E
6	F

$$\Pi_{\textit{roll\_no, name}} (\sigma_{\text{age} > 17} (\text{Student}))$$

Query 4

Display (or project) the roll number and name of the students whose age is greater than 17.

Set Operators in Relational Algebra



Set Operators in Relational Algebra

- Set operators: Union, intersection and set difference are binary operators.
  - ✓ They perform operation between two relation.

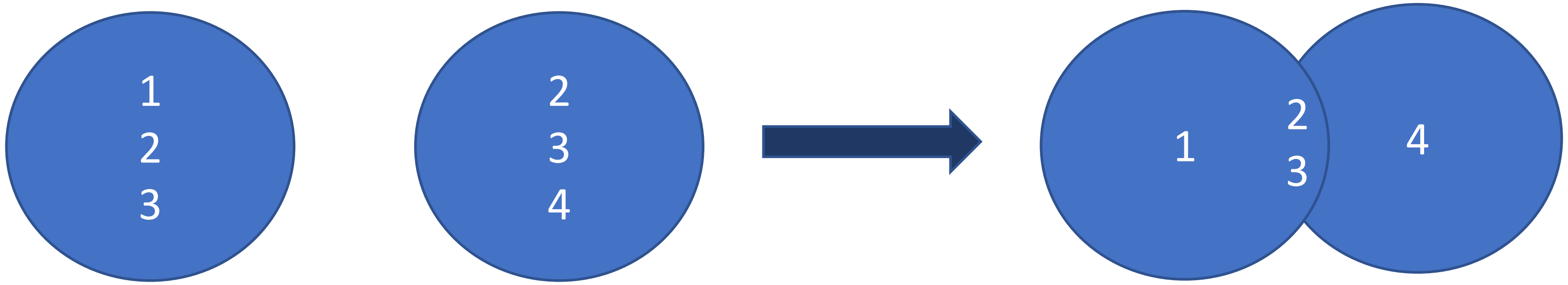
- To use set operator on two relations,
  - ✓ The two relation must be compatible.

- Two relations are compatible if,
  - Both the relation must have same number of attributes or columns.
  - Corresponding attribute or column have the same domain or type.

- Duplicates tuples are automatically eliminated.

Union Operator (∪)

- Suppose R and S are two relations, then ***Union operation*** selects all the tuples that are either in relations R or S or in both relations R and S.
- It eliminates the duplicate tuples.
- For a union operation to be valid, the following conditions must hold:
  - ✓ 1. Two relation R and S both have same number of attributes.
  - ✓ 2. Corresponding attribute or column have the same domain or type.
    - The attributes of R and S must occur in the same order.
  - ✓ 3. Duplicate tuple should automatically removed



Union Operator (∪)

- Symbol: ∪
- Notation: R ∪ S
- RA: R ∪ S
- SQL: (SELECT * FROM R) *UNION* (SELECT * FROM S)

Example: Union Operator (∪)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Student ∪ Employee

Roll_No	Name
1	A
2	B
3	C
4	D
8	G
9	H

Note: **Union is commutative: R ∪ S = S ∪ R**

Example: Union Operator (∪)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Name
A
B
C
D
G
H

$$\prod_{Name} (\text{Student}) \cup \prod_{Name} (\text{Employee})$$

Examples : Union Operator (∪)

- Find the names of author who have either written a book or an article or both;

$$\prod_{Author} (Books) \cup \prod_{Author} (Articles)$$

- Select all the information from two relation "Books" and "Authors" where subject code is "CSE 301" and price is less than 500.

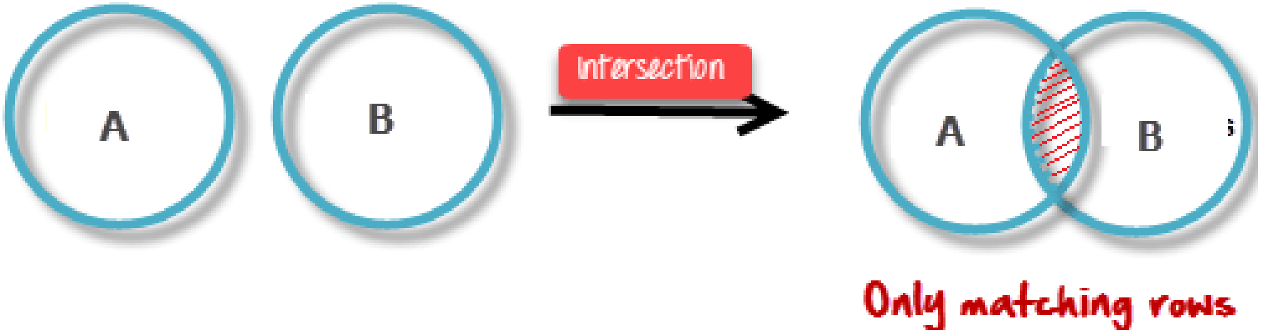
$$\prod (\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (Books)) \cup \prod (\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (Authors))$$

OR

$$(\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (Books)) \cup (\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (Authors))$$

Intersection Operator (∩)

- ❑ Defines a relation consisting of a set of all tuple that are in both A and B.
- ❑ A and B must be union-compatible.
- ❑ For a union operation to be valid, the following conditions must hold:
  - ✓ 1. Two relation R and S both have same number of attributes.
  - ✓ 2. Corresponding attribute or column have the same domain or type.
    - The attributes of R and S must occur in the same order.
  - ✓ 3. Duplicate tuple should automatically removed



Intersection Operator (∩)

- Symbol: ∩
- Notation: R ∩ S
- RA: R ∩ S
- SQL: (SELECT * FROM R) *INTERSECT* (SELECT * FROM S)

Example: Intersection Operator (∩)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Student ∩ Employee

Roll_No	Name
2	B

Note: **Intersection is commutative: R ∩ S = S ∩ R**

Example: Intersection Operator (∩)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Name
B

$$\prod_{Name} (\textbf{Student}) \cap \prod_{Name} (\textbf{Employee})$$

Examples: Intersection Operator (∩)

- Find the names of author who have written a book and an article both;

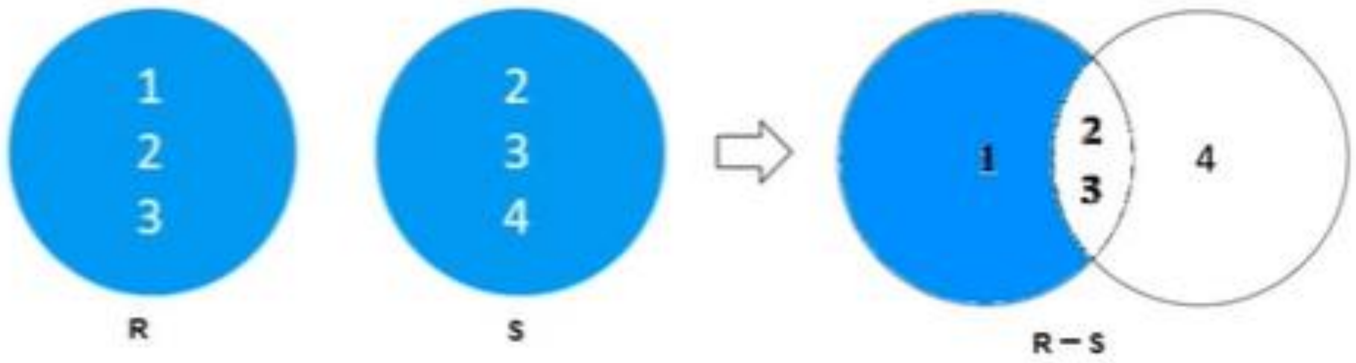
$$\prod_{Author} (\textbf{Books}) \cap \prod_{Author} (\textbf{Articles})$$

- Select the names of all the books from the two relationships "Books" and "Authors" that are present in both relationships, where the subject code is "CSE 301" and the price is less than 500.

$$\prod_{Book\_Name} (\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (\textbf{Books}))$$
$$\cap \prod_{Book\_Name} (\sigma_{Subject = \text{“CSE 301”} \wedge price < 500} (\textbf{Authors}))$$

Set Difference (-)

- The result of R – S, is a relation which includes all tuples that are in first relation (R) but not in second relation (S).
- A and B must be union-compatible.
- For a union operation to be valid, the following conditions must hold:
  - ✓ 1. Two relation R and S both have same number of attributes.
  - ✓ 2. Corresponding attribute or column have the same domain or type.
    - The attributes of R and S must occur in the same order.



Set Difference (-)

- Symbol: -
- Syntax: R - S
- RA: R - S
- SQL: (SELECT * FROM R) *EXCEPT* (SELECT * FROM S)

Example: Set Difference (-)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Student - Employee

Roll_No	Name
1	A
3	C
4	D

Note: **1. Set Difference is not commutative: R - S ≠ S – R**

2. R – (R – S) = R ∩ S

Intersection can be derived from set difference that’s why intersection is a derived operator

Example: Set Difference (-)

Student

Roll_No	Name
1	A
2	B
3	C
4	D

Employee

Emp_id	Name
2	B
8	G
9	H

Name
A
C
D

$\prod_{Name}$ **(Student) - $\prod_{Name}$ (Employee)**

Examples: Set Difference (-)

- Find the names of author who have written books but not article;

$$\prod_{Author} (Books) - \prod_{Author} (Articles)$$

- Select the names of all the books from the two relationships “Shope1" and " Shope2" that are present in " Shope1" relationship but not in “Shope2”, where the subject code is "CSE 301".

$$\prod_{Book\_Name} (\sigma_{Subject = \text{“CSE 301”}} (Shope1)) - \prod_{Book\_Name} (\sigma_{Subject = \text{“CSE 301”}} (Shope2))$$

Cartesian Product (X)

- It is a fundamental operator in relational algebra
- Cartesian product combines information of two different relations into one
- It is also called Cross product
  - Generally a Cartesian product is never a meaningful operation when it is performed alone. However, it becomes meaningful when it is followed by other operations.
  - Generally it is followed by select operations.

Cartesian Product (X)

- Symbol: X
- Syntax: R X S
- RA: R X S
- SQL: SELECT * FROM R, S

Example: Cartesian Product (X)

Student

Roll_No	Name
1	A
2	B

Course

Emp_id	Std_Name	Age
CSE 101	A	10
CSE 102	G	19
CSE 103	H	25
CSE 101	E	21

Student X Employee

Roll_No	Name	Emp_id	Std_Name	Age
1	A	CSE 101	A	10
1	A	CSE 102	G	19
1	A	CSE 103	H	25
1	A	CSE 101	E	21
2	B	CSE 101	A	10
2	B	CSE 102	G	19
2	B	CSE 103	H	25
2	B	CSE 101	E	21

Note: **Cartesian Product is commutative: R X S = S X R**

Cartesian Product (X): Rules

- If relation R and S have a and b attributes respectively, then resulting relation will have (a + b) attributes from both the input relations.

- If relation r1 and R2 have n1 and n2 tuples respectively, then resulting relation will have (n1 x n2) tuples, combining each possible pair of tuples from both the relations.

- If both input relation have some attributes having same name, change the name of the attribute with the name of the relation "realation_name.attribute_name"

Cartesian Product (X)

- If both input relation have some attributes having same name, change the name of the attribute with the name of the relation "realation_name.attribute_name"

R

A	B
1	A
2	B

S

B	C	D
2	B	10
8	G	19
9	H	25
5	E	21

R X S

A	R . B	S . B	C	D
1	A	2	B	10
1	A	8	G	19
1	A	9	H	25
1	A	5	E	21
2	B	2	B	10
2	B	8	G	19
2	B	9	H	25
2	B	5	E	21

Example: Cartesian Product (X)

R

A	B
1	A
2	B

S

C	D
2	X
8	G
9	H
6	F

R X S

A	B	C	D
1	A	2	X
1	A	8	G
1	A	9	H
1	A	6	F
2	B	2	X
2	B	8	G
2	B	9	H
2	B	6	F

A	B	C	D
2	B	2	X

$\sigma_{A = C}$ (R X S)

Example: Cartesian Product (X)

S

Roll_No	Name
1	A
2	B
3	C
4	D

E

Emp_id	Name
2	B
3	G
9	H
6	F
5	K

Roll_No	S.Name	Emp_id	E.Name
2	B	2	B
3	C	3	G

$\sigma_{\text{Roll\_No} = \text{Emp\_id}} (S \times E)$

Example: Cartesian Product (X)

S

Roll_No	Name
1	A
2	B
3	C
4	D

E

Emp_id	Name
2	B
3	G
9	H
6	C
5	K

Roll_No	S.Name	Emp_id	E.Name
2	B	2	B
3	C	6	C

$\sigma_{\text{S.Name} = \text{E.Name}}(S \times E)$

Example: Cartesian Product (X)

S

Roll_No	Name
1	A
2	B
3	C
4	D

E

Emp_id	Name
2	B
3	G
9	H
6	C
5	K

Emp_id
2
6

$$\prod_{Emp\_id}(\sigma_{S.Name\ =\ E.Name}(S \times E))$$

Example: Cartesian Product (X)

- Find the books and articles written by Korth.

$$\sigma_{\textit{Author} = \text{“korth”}}(\text{Books} \times \text{Articles})$$