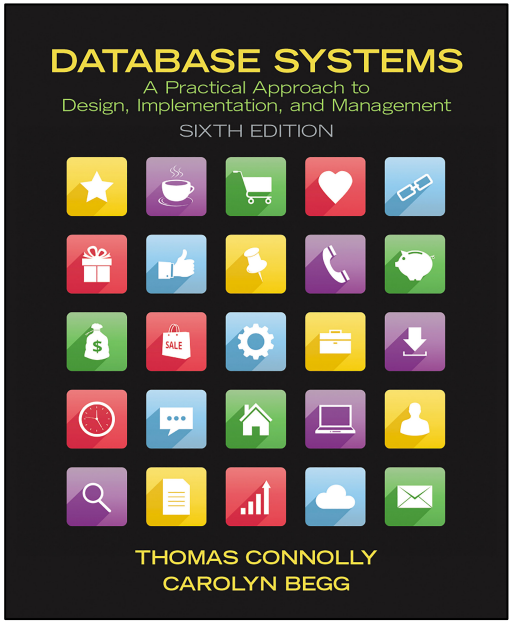
Relational Algebra

Topic 2
Lesson 9 – Relational Algebra

Chapter 5 section 1 Connolly and Begg



What is Relational Algebra?

What is an algebra?

A mathematical system consisting of:
Operands --- variables or values from which new values can be constructed.
Operators --- symbols denoting procedures that construct new values from given values.

What is relational algebra?

Operands are relations or variables that represent a relation
Operators are designed to do the most common things that we need to do with relations in a database.

Properties of relational algebra

Relational algebra operations work on one or more relations to define another relation without changing the original relations.

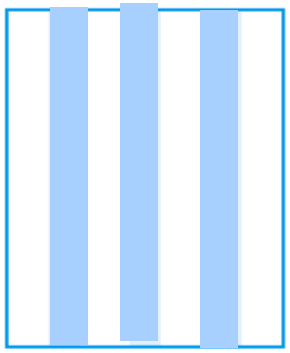
Both operands and results are relations, so output from one operation can become input to another operation.

Allows expressions to be nested, just as in arithmetic. This property is the **closure property.**

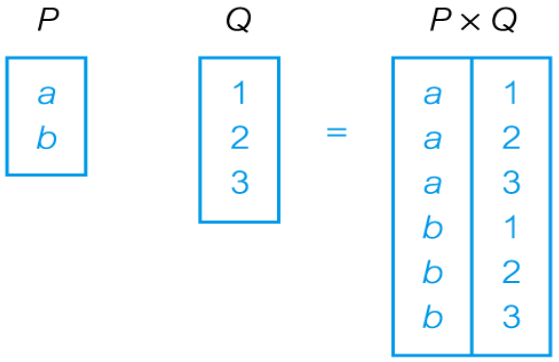
Relational Algebra: a collection of operations that users can perform on relations to obtain a desired result

Basic Operations of Relational Algebra

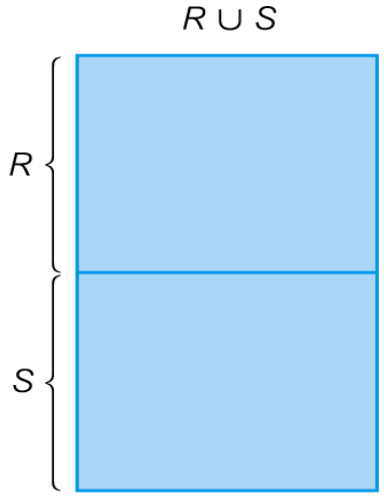
(a) Selection



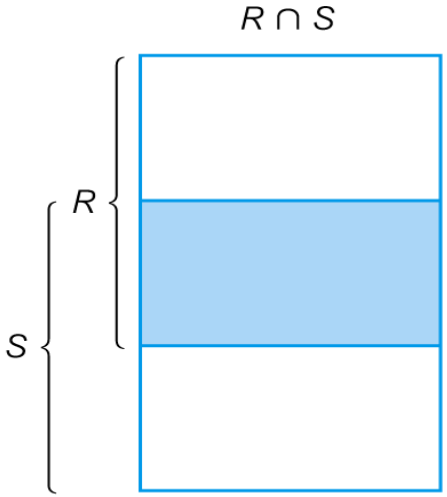
(b) Projection



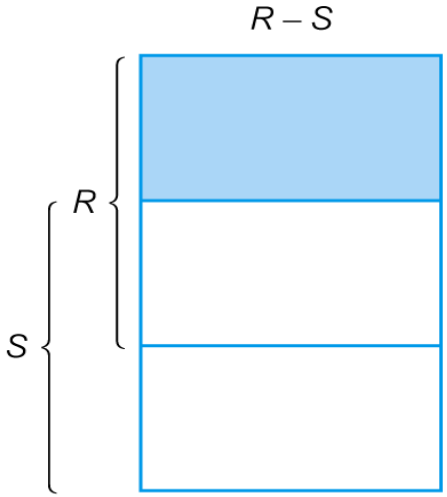
(c) Cartesian product



(d) Union



(e) Intersection



(f) Set difference

Basic operations

- SELECT : filter by rows
- PROJECT: filter by columns
- UNION: combine n compatible relations into one relation
- INTERSECT: of relation A and B
  are the tuples in both A and B
- SET DIFFERENCE: (A-B) tuples in A but not in B
- CARTESIAN PRODUCT: (AxB) each tuple in A is concatenated with each tuple in B

Selection

Provides a filter to the tuples in the relation R. The operation is represented via the **σ** (Greek letter sigma). AND (∧) , OR (V) and NOT (∽) are supported for creating compound predicates.

$$\sigma_{predicate}(R)$$

```
SELECT * FROM student where
id = 3;
```

Example:

$$\sigma_{(id = 3)}(student)$$

RESULT

id	Name	School	Credits_Earned	Credits_Req	Yr_grad
3	Li	Khoury	50	120	2020

SQL representation?

Projection

Provides a filter to the columns in the relation R. It defines a result which is a subset of the relation R. The operation is represented via the Π (Greek letter pi)

$$\Pi_{col1, ..., coln} (R)$$

Example:

Π id, name, school (student)

RESULT

sid	Name	School
1	Smith	Khoury
2	Shah	D'Amore McKim
3	Li	Khoury

SQL representation?

```
SELECT id, name, school FROM student;
```

Union

- Union of two relations R and S defines a relation that contains all the tuples of R, as well as the tuples of S, duplicate tuples are eliminated.
- R and S must be union-compatible.
- If R and S have *I* and *J* tuples, respectively, union is obtained by concatenating them into one relation with a maximum of (*I* + *J*) tuples.

SYNTAX: $R \cup S$

Example:
(student1) U (student2)

UNION Result

Example:
(student1) U (student2)

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
3	Li	Khoury	50	120	2020
1	Smith	Khoury	32	120	2019

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019
3	Li	Khoury	50	120	2020

Set Difference

- Identifies the tuples that are in R but not in S
- R and S must be union-compatible.

SYNTAX: $R - S$

Example:

$\Pi_{\text{id, name, school}}(\text{student1}) - \Pi_{\text{id, name, school}}(\text{student2})$

Set Difference Example

Example:

$\Pi_{\text{id, name, school}}(\text{student3}) - \Pi_{\text{id, name, school}}(\text{student2})$

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019
3	Li	Khoury	50	120	2020

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
3	Li	Khoury	50	120	2020

sid	Name	School
1	Smith	Khoury

Intersection

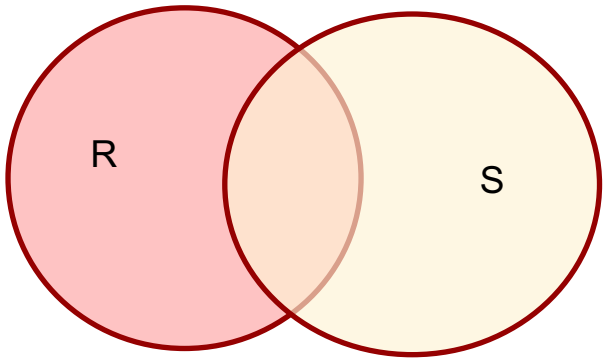
- Identifies the tuples that are in R as well as in S
- R and S must be union-compatible.
- Can be derived from set difference $R \cap S = R - (R - S)$

SYNTAX:

$$R \cap S$$

Example:

$$\Pi_{\text{id, name, school}}(\text{student1}) \cap \Pi_{\text{id, name, school}}(\text{student2})$$



Example: Intersection

Example:

$\Pi_{\text{id, name, school}}(\text{student3}) \cap \Pi_{\text{id, name, school}}(\text{student2})$

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019
3	Li	Khoury	50	120	2020

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
3	Li	Khoury	50	120	2020

sid	Name	School
3	Li	Khoury

Cartesian Product

- Defines a relation that is the concatenation of every tuple of relation R with every tuple of relation S.

SYNTAX: $R \times S$

Example:

$\Pi_{\text{id, name, school}}$ (student1) X (student_major)

SID	MID
1	1
1	3
2	1
3	2

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019
3	Li	Khoury	50	120	2020

Result Cartesian Product

Student.sid	Name	School	Student_major.sid	mid
1	Smith	Khoury	1	1
1	Smith	Khoury	1	3
1	Smith	Khoury	2	1
1	Smith	Khoury	3	2
3	Li	Khoury	1	1
3	Li	Khoury	1	3
3	Li	Khoury	2	1
3	Li	Khoury	3	2

Example: Cartesian Product

Example:

$\Pi_{\text{id, name, school}}$ (student1) X (available_major)

sid	Name	School	Credits_Earned	Credits_Req	Yr_grad
1	Smith	Khoury	32	120	2019
3	Li	Khoury	50	120	2020

Can I do a cartesian product between relations with no relationship ?

sid	Name	School
1	Smith	Khoury
3	Li	Khoury

X

mid	major
1	CS
2	DS
3	Accounting

Result Cartesian Product

Yes – but the results are meaningless.

sid	Name	School	mid	Major
1	Smith	Khoury	1	CS
1	Smith	Khoury	2	DS
1	Smith	Khoury	3	Accounting
3	Li	Khoury	1	CS
3	Li	Khoury	2	DS
3	Li	Khoury	3	Accounting

RA and SQL practice work (1)

Select the id, name, school from student1 and the student2 table

$$\Pi_{\text{id, name, school}}(\text{student1}) \cup \Pi_{\text{id, name, school}}(\text{student2})$$

Select the id, name, school from student for the student with id 3.

$$\Pi_{\text{id, name, school}}\, \sigma_{(\text{id} = 3)}(\text{student})$$

Select the id, name, school from student for the students with id 3 or id 2.

$$\sigma_{(\text{id} = 3 \ \vee \ \text{id} = 2)}\, \Pi_{\text{id, name, school}}(\text{student})$$

RA and SQL practice work (2)

Select the id, name, school from student where the name is Smith and the School name is Khoury

$$\sigma_{(\text{school} = \text{'Khoury'} \wedge \text{name} = \text{'Smith'})} \Pi_{\text{id, name, school}} (\text{student})$$

$$\sigma_{(\text{id} = 3 \text{ AND school} = \text{'Khoury'})} \Pi_{\text{id, name, school}} (\text{student})$$

Select the id, name, school from student1 as well as the id, name, school from student2 table who are not found in the student3 table

$$\Pi_{\text{id, name, school}} (\text{student1}) \cup (\Pi_{\text{id, name, school}} (\text{student2}) - \Pi_{\text{id, name, school}} (\text{student3}))$$

RA for examples

$\Pi_{\text{id, name, school}}(\text{student1}) \cup \Pi_{\text{id, name, school}}(\text{student2})$

$\Pi_{\text{id, name, school}}\, \sigma_{(\text{id} = 3)}(\text{student})$

$\sigma_{(\text{id} = 3 \ \vee \ \text{id} = 2\ )}\, \Pi_{\text{id, name, school}}(\text{student})$

$\sigma_{(\text{school} = \text{'Khoury'} \ \wedge \ \text{name} = \text{'Smith'}\ )}\, \Pi_{\text{id, name, school}}(\text{student})$

$\sigma_{(\text{id} = 3 \text{ AND school} = \text{'Khoury'})}\, \Pi_{\text{id, name, school}}(\text{student})$

$$\Pi_{\text{id, name, school}}(\text{student1}) \cup (\ \Pi_{\text{id, name, school}}(\text{student2}) - \Pi_{\text{id, name, school}}(\text{student3})\ )$$

Can you convert these RA expressions to SQL?

SQL representation (1)

$\Pi_{\text{id, name, school}}(\text{student1}) \cup \Pi_{\text{id, name, school}}(\text{student2})$

SELECT id, name, school FROM student1
UNION
SELECT id, name, school FROM student2;

$\Pi_{\text{id, name, school}}\ \sigma_{(\text{id} = 3)}(\text{student})$

SELECT id, name, school FROM student WHERE id = 3;

$\sigma_{(\text{id} = 3\ \vee\ \text{id} = 2\ )}\ \Pi_{\text{id, name, school}}(\text{student})$

SELECT id, name, school FROM student
WHERE id = 3 OR id = 2;

$\sigma_{(\text{school} = \text{‘Khoury’}\ \wedge\ \text{name} = \text{‘Smith’}\ )}\ \Pi_{\text{id, name, school}}(\text{student})$

SELECT id, name, school FROM student
WHERE name = ‘Smith’ AND school = ‘Khoury’;

SQL representation (2)

$\sigma_{(id = 3 \wedge school = \text{'Khoury'})} \Pi_{id, name, school} (student)$

SELECT id, name, school FROM student WHERE id = 3
AND school = 'Khoury' ;

$\Pi_{id, name, school} (student1) \cup$
$(\Pi_{id, name, school} (student2)$
$- \Pi_{id, name, school} (student3))$

SELECT id, name, school FROM student1
UNION
SELECT id, name, school FROM student2
WHERE id, name, school NOT IN
(SELECT id, name, school FROM student3) ;

Summary

- SELECT : filter by rows
- PROJECT: filter by columns
- UNION: combine n compatible relations into one relation
- INTERSECT: of relation A and B
  are the tuples in both A and B
- SET DIFFERENCE: (A-B) tuples in A but not in B
- CARTESIAN PRODUCT: (AxB) each tuple in A is concatenated with each tuple in B