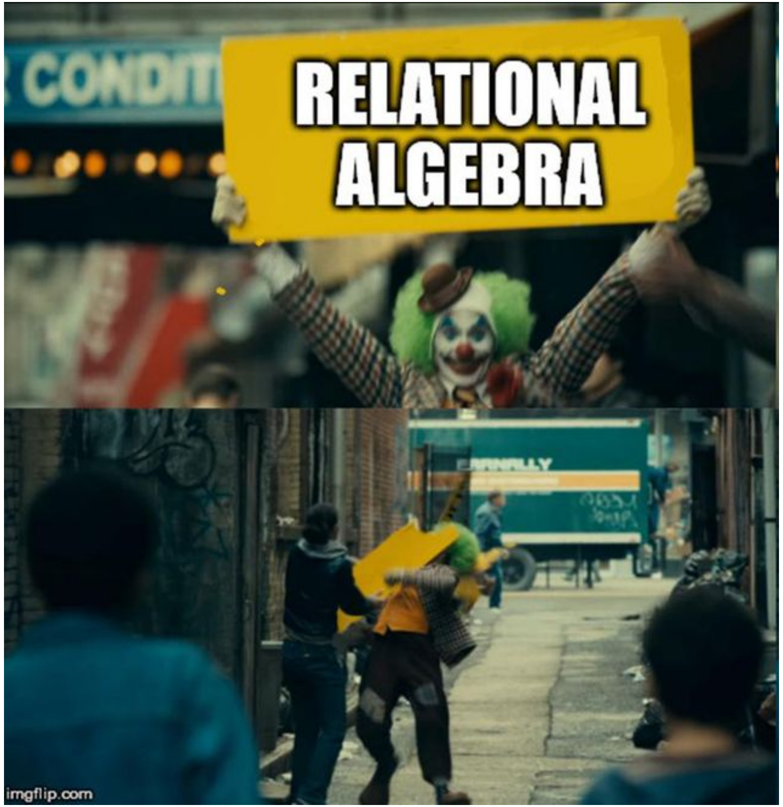
Databases - Tutorial 04
Relational Algebra

Hamza Salem - Innopolis University

Contents

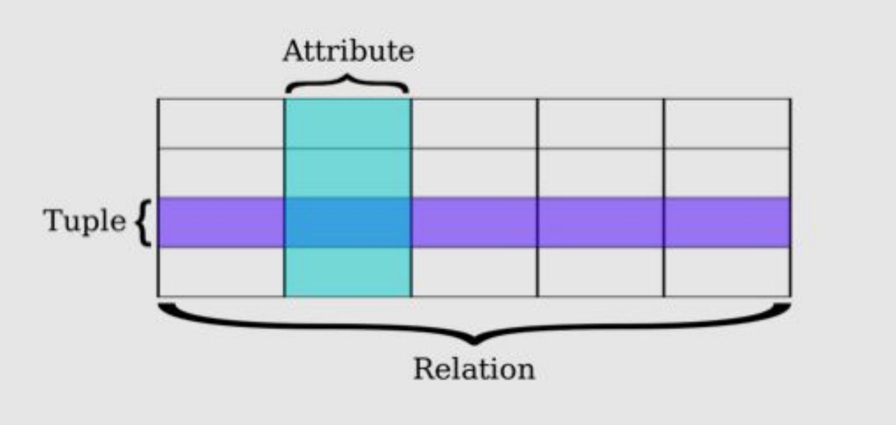
- Relational Algebra



Enhanced ERD

A relation is a set of tuples (d1, d2, ..., dn), where each element dj is a member of Dj, a data domain (all the values which a data element may contain)

- No ordering to the elements of the tuples of a relation
- Relation, tuple, and attribute are commonly represented as table, row, and column respectively



Relations

Relations are sets, so we can apply set-theoretic operators + special relational operators

Basic operators

1) Union: ∪

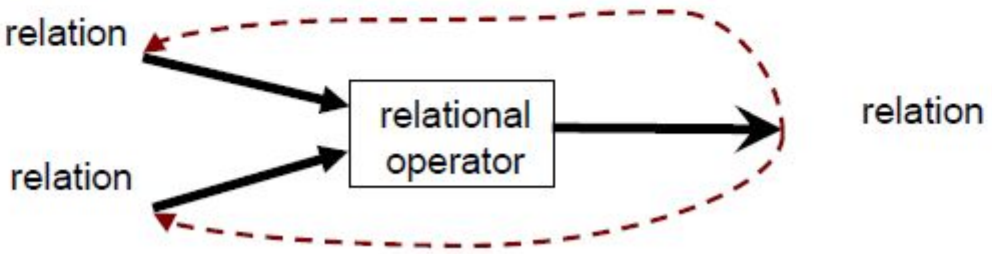
2) Set difference: –

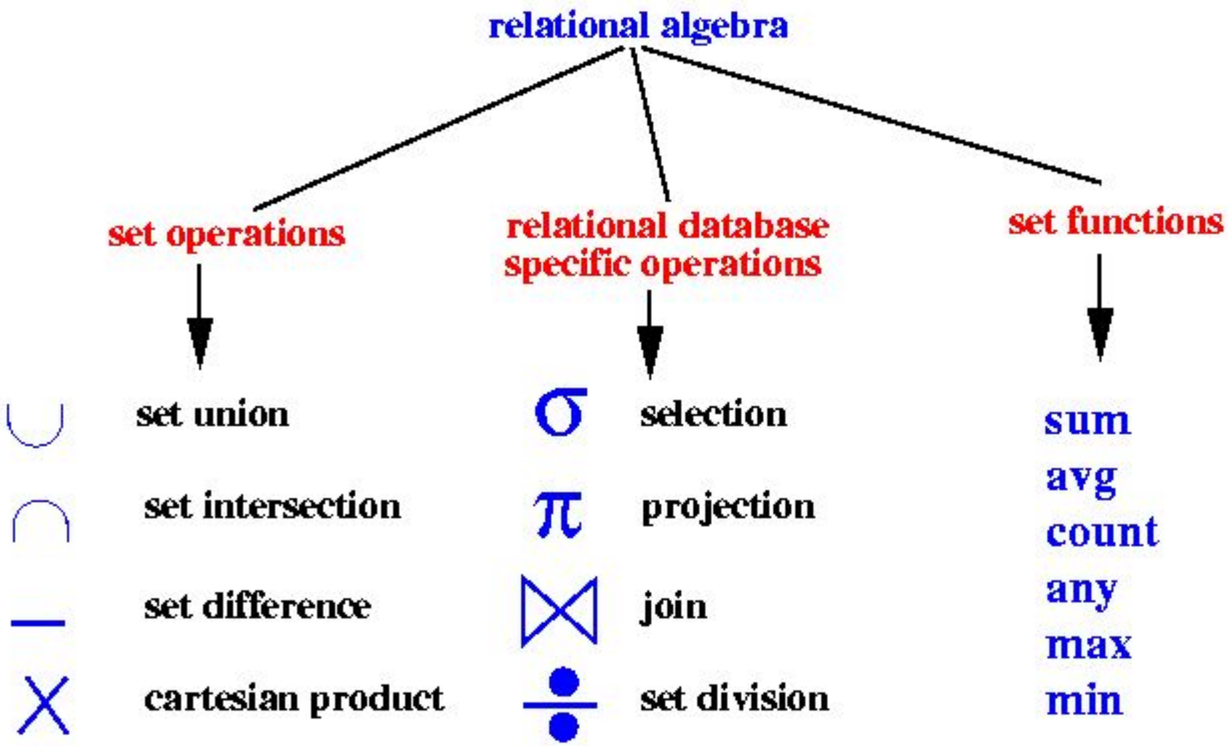
3) Cartesian product: x

4) Select: σ

5) Project: Π

Also Rename, Intersection, Join and Division…




relational algebra
set operations
relational database specific operations
set functions
set union
set intersection
set difference
cartesian product
selection
projection
join
set division
sum
avg
count
any
max
min

Operators


Unary operators perform an action with a single operand. Binary operators perform actions with two operands.
UNTIL WE MEET AGAIN.
imgflip.com

Union

Binary operator

Tuples in relation 1 OR in relation 2

Tuples must be union-compatible

- Same number of columns (attributes)
- 'Corresponding' columns have the same domain (type)

Eliminates duplicates

Notation: R1 ∪ R2

ID	Firstname	Lastname
125	John	Smith
214	Anna	Kim
336	Leo	Abel

Attends course 1

ID	Firstname	Lastname
231	Maria	Dawn
214	Anna	Kim
255	Jim	White

Attends course 2

Attends course 1 or 2

ID	Firstname	Lastname
125	John	Smith
214	Anna	Kim
336	Leo	Abel
231	Maria	Dawn
255	Jim	White

Set Difference

Binary operator

Tuples in relation 1 AND NOT in relation 2

Tuples must be union-compatible

- Same number of columns (attributes)
- `Corresponding' columns have the same domain (type)

Non-commutative

Notation: R1-R2 or R1\R2

Set difference (keep the tuples that are in relation 1, but not in relations 2 (binary))

Students

ID	Firstname	Lastname
125	John	Smith
214	Anna	Kim
336	Leo	Abel

Didn't graduate

ID	Firstname	Lastname
231	Maria	Dawn
214	Anna	Kim
255	Jim	White

Graduated students

ID	Firstname	Lastname
125	John	Smith
336	Leo	Abel

$$\{1, 2, 3\} \setminus \{2, 3, 4\} = \{1\}.$$

$$\{2, 3, 4\} \setminus \{1, 2, 3\} = \{4\}.$$

Intersection

Binary operator

Tuples in relation 1 AND in relation 2

Tuples must be union-compatible

- Same number of columns (attributes)
- `Corresponding' columns have the same domain (type)

commutative

Notation: R1∩R2

Intersection (keep the tuples that are in relation 1 AND in relation 2 (binary))

Master students

ID	Firstname	Lastname
125	John	Smith
214	Anna	Kim
336	Leo	Abel

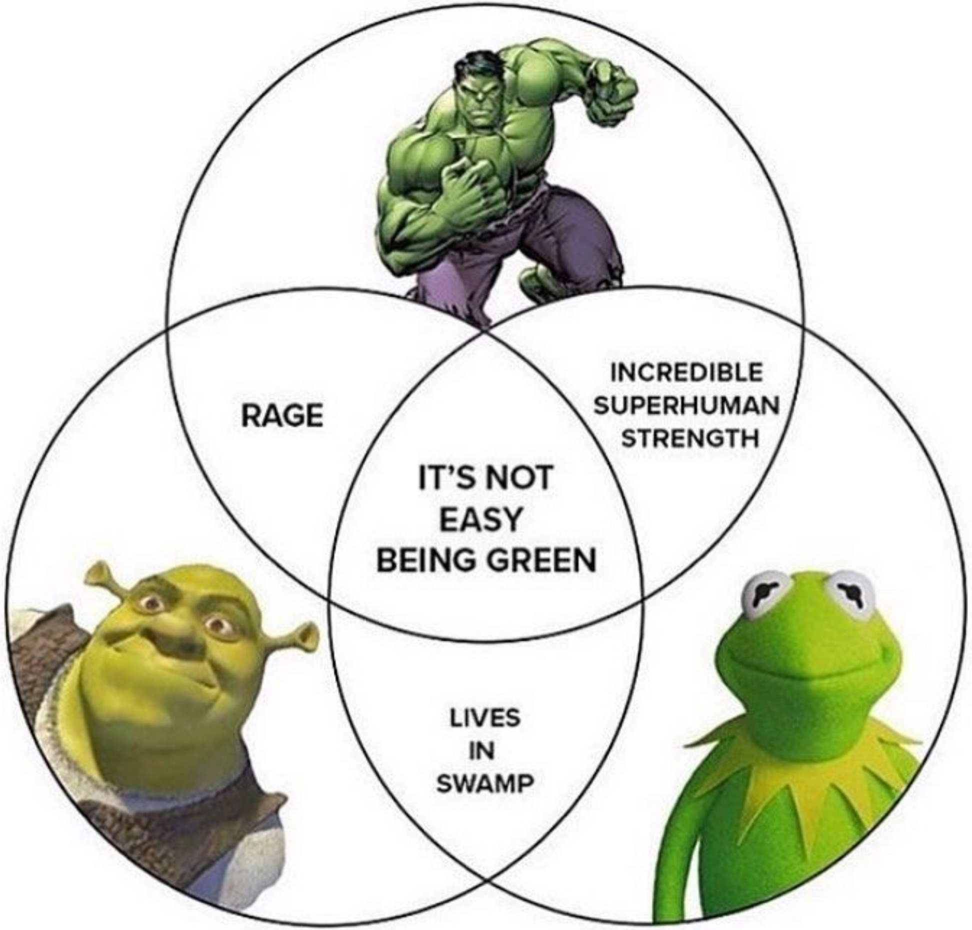
Graduated students

ID	Firstname	Lastname
231	Maria	Dawn
214	Anna	Kim
255	Jim	White

Graduated master students

ID	Firstname	Lastname
214	Anna	Kim

Hulk ___ Shrek =Rage
Hulk ___ Kermit =Rage



Cartesian product

- S1 X R1: Each row of S1 paired with each row of R1
- Like the cartesian product for mathematical relations
- Every tuple of S1 “appended” to every tuple of R1
- How many rows in the result?
- No need for the two input relations to be union-compatible
- Result schema has one attribute per attribute of S1 and R1

Notation: S1xR1

Students

ID	Firstname	Lastname
125	John	Smith
214	Anna	Kim
336	Leo	Abel

Courses

CID	Course
11	Logic
12	DB

Courses x Students

ID	Firstname	Lastname	CID	Course
125	John	Smith	11	Logic
214	Anna	Kim	11	Logic
336	Leo	Abel	11	Logic
125	John	Smith	12	DB
214	Anna	Kim	12	DB
336	Leo	Abel	12	DB

Renaming

The problem: Father and Mother are different names, but both represent a parent.

The solution: rename attributes!

Paternity

Father	Child
John	Igor
Jim	Eva
Leo	Kate

Maternity

Mother	Child
Anna	Kate
Maria	Igor
Elena	Andrew

Renaming

Rename

- Unary operator
- Changes attribute **names** for a relation without changing any values
- Renaming removes the limitations associated with set operators

Notation: ρOldName➔NewName(r) (e.g ρFather➔Parent(Paternity))

- If there are two or more attributes involved in a renaming operation, then ordering is

meaningful: (e.g., ρBranch,Salary ➔ Location,Pay(Employees))

Paternity

Father	Child
John	Igor
Jim	Eva
Leo	Kate

$\rho Father \rightarrow Parent(Paternity)$

Parent	Child
John	Igor
Jim	Eva
Leo	Kate

Maternity

Mother	Child
Anna	Kate
Maria	Igor
Elena	Andrew

$\rho Mother \rightarrow Parent(Paternity)$

Parent	Child
Anna	Kate
Maria	Igor
Elena	Andrew

Select

- Unary operator
- Selects a subset of rows from a relation that satisfy selection predicate
- Schema of result is same as that of the input relation
- Works like a filter that keeps only those tuples that satisfy a qualifying condition
- The selection condition is a Boolean expression specified on the attributes of relation R

Notation: σp(r)

Select Example: $\sigma_{Age > 20}$ **(Students)**

Students

ID	Firstname	Lastname	Age
125	John	Smith	21
214	Anna	Kim	19
336	Leo	Abel	22
231	Maria	Dawn	18
255	Jim	White	23

Students with age > 20

ID	Firstname	Lastname	Age
125	John	Smith	21
336	Leo	Abel	22
255	Jim	White	23

How to select students with age greater than 20 and GPA greater than 3.2?

Students

ID	Firstname	Lastname	Age	GPA
125	John	Smith	21	3.1
214	Anna	Kim	19	3.84
336	Leo	Abel	22	3.69
231	Maria	Dawn	18	3.21
255	Jim	White	23	2.9

Projection

- Unary operator
- Deletes unwanted columns from a relation
- Removes duplicated data
- The schema of result has exactly the columns in the projection list, with the same names

that they had in the input relation

Notation: Пp(r)

Projection example: $\Pi_{\text{Lastname,Age}}$ **(Students)**

Students

ID	Firstname	Lastname	Age
125	John	Smith	21
214	Anna	Kim	19
336	Leo	Abel	22
231	Maria	Dawn	18
255	Jim	Smith	21

$\Pi_{\text{Lastname,Age}}$ (Students)

Lastname	Age
Smith	21
Kim	19
Abel	22
Dawn	18

Extended projection example: $\Pi_{\text{Firstname + Lastname->Name,Age}}$ **(Students)**

Students

ID	Firstname	Lastname	Age
125	John	Smith	21
214	Anna	Kim	19
336	Leo	Abel	22
231	Maria	Dawn	18
255	Jim	Smith	21

Projected table

Name	Age
John Smith	21
Anna Kim	19
Leo Abel	22
Jim Dawn	18
Jim Smith	21

Join

- Binary operator
- Allows us to establish connections among data in different relations, taking advantage of

the "value-based" nature of the relational model

- Two versions
  - **"natural" join: takes attribute names into account**
  - **"theta" join.**

Notation: $r1 \bowtie r2$

Natural join (or "just join")

- Binary operator
- Select rows where attributes that appear in both relations have equal values
- Project all unique attributes and one copy of each of the common ones

Notation: R ⋈ S

Attendance

FirstName	Lastname	Course
John	Smith	Logic
John	Smith	DB
Leo	Abel	DB

Courses

CID	Course	Teacher
11	Logic	Pain
12	DB	White
13	English	Gray

Attendance ⋈ Courses

Firstname	Lastname	Course	CID	Teacher
John	Smith	Logic	11	Pain
John	Smith	DB	12	White
Leo	Abel	DB	12	White

Note: Joins can be incomplete or empty

Theta join (or “conditional join”)

- Binary operator

- Results in all combinations of tuples in R and S that satisfy θ (where θ is a binary relational

operator in the set {<, ≤, =, >, ≥})

- Result schema same as that of cross-product

- In case the operator θ is the equality operator (=) then this join is also called an equijoin

Notation: $R \bowtie_\theta S = \sigma_\theta(R \times S)$

Group A

Lastname	**Age**
Smith	20
Kim	32
Abel	17

Group B

Lastname	**Age**
White	21
Gray	32
Li	17

Group A $\bowtie_{A.Age>B.Age}$ Group B

Lastname	**Age**	**Lastname**	**Age**
Kim	32	White	21
Smith	20	Li	17
Kim	32	Li	17

Equijoin

- In case the operato θ is the equality operator (=) then this join is also called an equijoin

Students

ID	Lastname	Project
125	Smith	Moon
214	Kim	Solar
336	Abel	Solar

Projects

CID	Name
11	Solar
12	Moon

Students $\bowtie_{Project=Name}$ Projects

ID	Lastname	Project	CID	Course
125	Smith	Moon	12	Moon
214	Kim	Solar	11	Solar
336	Abel	Solar	11	Solar

Division

The division operator is used for queries which involve the ‘all’.

R1 ÷ R2 = tuples of R1 associated with all tuples of R2.

R

ColA	ColB
F	1
F	2
F	3
E	1
E	3
S	1
S	2

S

ColB
1
2

$R \div S =$

ColA
F
S

Let us try together

- Suppliers (<u>sid: integer</u>, sname: string, address: string)
- Parts (<u>pid: integer</u>, pname: string, color: string)
- Catalog (<u>sid: integer, pid: integer</u>, cost: real)

1- Find the sids of suppliers who supply some red or green part.

2- Find the sids of suppliers who supply some red part and some green part.

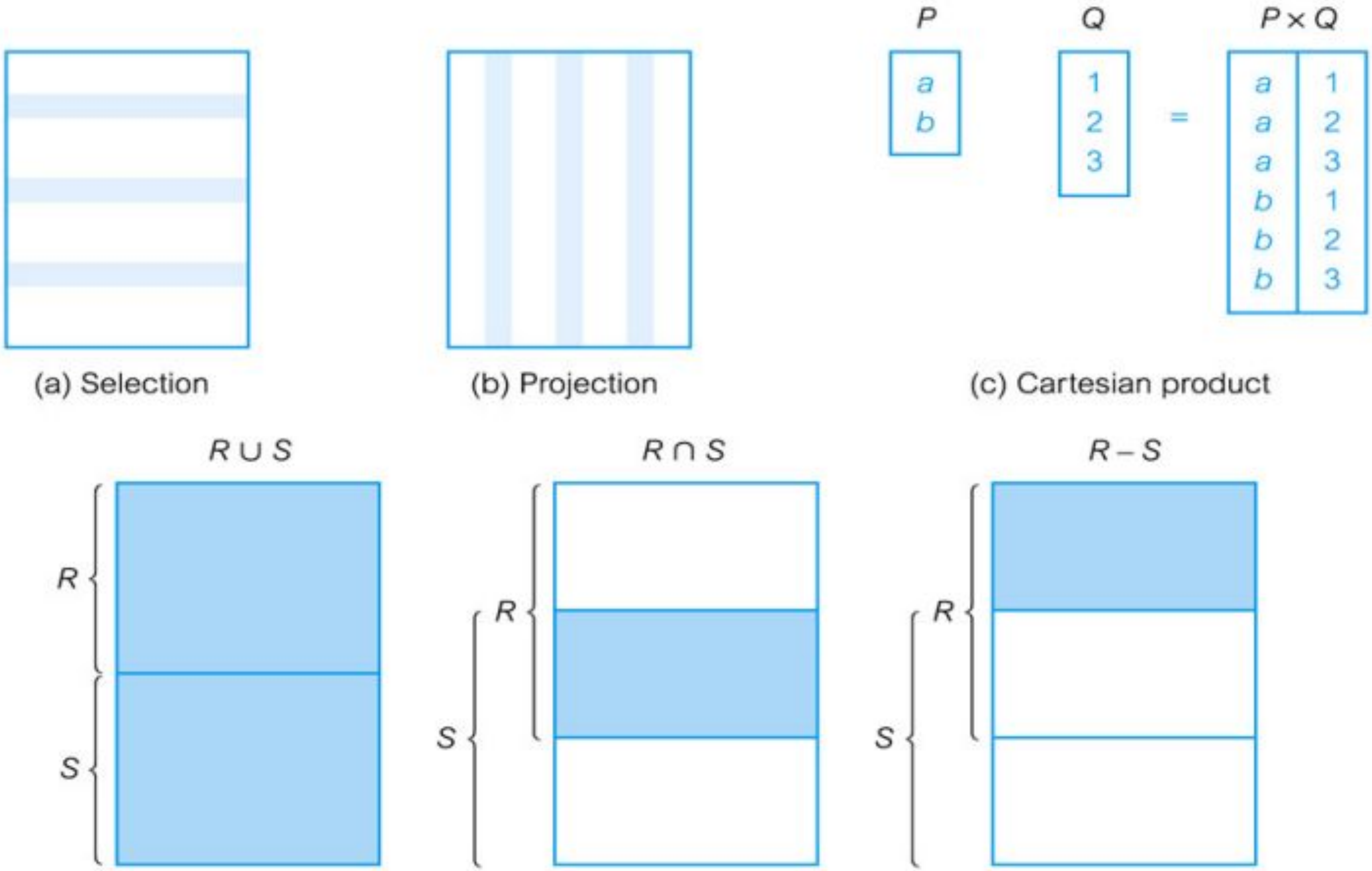
3- Find the sids of suppliers who supply every part.

$\pi_{sid}(\pi_{pid}(\sigma_{color='red' \lor color='green'} Parts) \bowtie catalog)$

$\rho(R1, \pi_{sid}((\pi_{pid}\, \sigma_{color='red'} Parts) \bowtie Catalog))$

$\rho(R2, \pi_{sid}((\pi_{pid}\, \sigma_{color='green'} Parts) \bowtie Catalog))$

$R1 \cap R2$



(a) Selection
(b) Projection
(c) Cartesian product
(d) Union
(e) Intersection
(f) Set difference

T

A	B
a	1
b	2

U

B	C
1	x
1	y
3	z

$T \bowtie U$

A	B	C
a	1	x
a	1	y

(g) Natural join

$T \triangleright_B U$

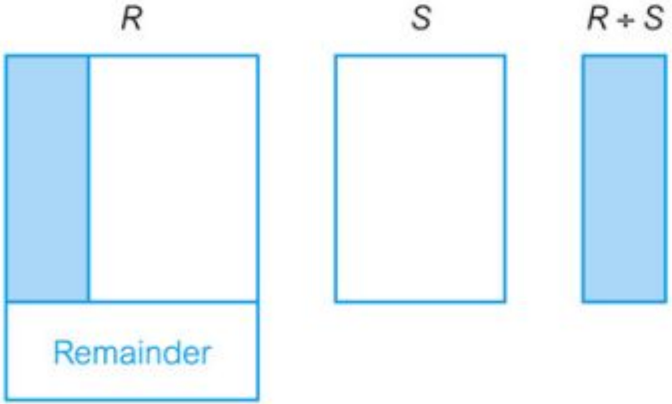
A	B
a	1

(h) Semijoin

$T ⟕_C U$

A	B	C
a	1	x
a	1	y
b	2	

(i) Left Outer join



(j) Divis on (shaded area)

V

A	B
a	1
a	2
b	1
b	2
c	1

W

B
1
2

$V \div W$

A
a
b

Example of division

Let us try together

- Suppliers (<u>sid: integer</u>, sname: string, address: string)
- Parts (<u>pid: integer</u>, pname: string, color: string)
- Catalog (<u>sid: integer, pid: integer</u>, cost: real)

1- Find the sids of suppliers who supply some red or green part.

2- Find the sids of suppliers who supply some red part and some green part.

3- **Find the sids of suppliers who supply every part.**

Parts

PID	Pname	Color
1	Red1	Red
2	Red2	Red
3	Green1	Green
4	Blue1	Blue
5	Red3	Red

Catalog

SID	PID	Cost
1	1	$10.00
1	2	$20.00
1	3	$30.00
1	4	$40.00
1	5	$50.00
2	1	$9.00
2	3	$34.00
2	5	$48.00
3	1	$11.00

SID	Sname	Address
1	Yosemite Sham	Devil's canyon, AZ
2	Wiley E. Coyote	RR Asylum, NV

Let us try together

- Suppliers (<u>sid: integer</u>, sname: string, address: string)
- Parts (<u>pid: integer</u>, pname: string, color: string)
- Catalog (<u>sid: integer, pid: integer</u>, cost: real)

$(\Pi_{sname}((\sigma_{color=red}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers)) \cap (\Pi_{sname}((\sigma_{color=green}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers))$

Sol : Find the Supplier names of the suppliers who supply a red part that costs less than 100 dollars and a green part that costs less than 100 dollars.

References

- https://www.guru99.com/relational-algebra-dbms.html
- https://www.javatpoint.com/dbms-relational-algebra
- https://home.adelphi.edu/~siegfried/cs443/443l9.pdf