ICT502/ITS571 RELATIONAL ALGEBRA

FAIQAH HAFIDZAH HALIM
MUHAMMAD HAMIZ MOHD RADZI

Lecture Content

- Introduction
- Relational Algebra
  - *Unary Operations*
  - *Set Operations*
  - *Join Operations*
  - *Division Operations*
  - *Aggregation & Grouping Operations*

Introduction

- 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 called closure.

Introduction

- Five basic operations in relational algebra: Selection, Projection, Cartesian product, Union, and Set Difference.
- These perform most of the data retrieval operations needed.
- Also have Join, Intersection, and Division operations, which can be expressed in terms of 5 basic operations.

Introduction

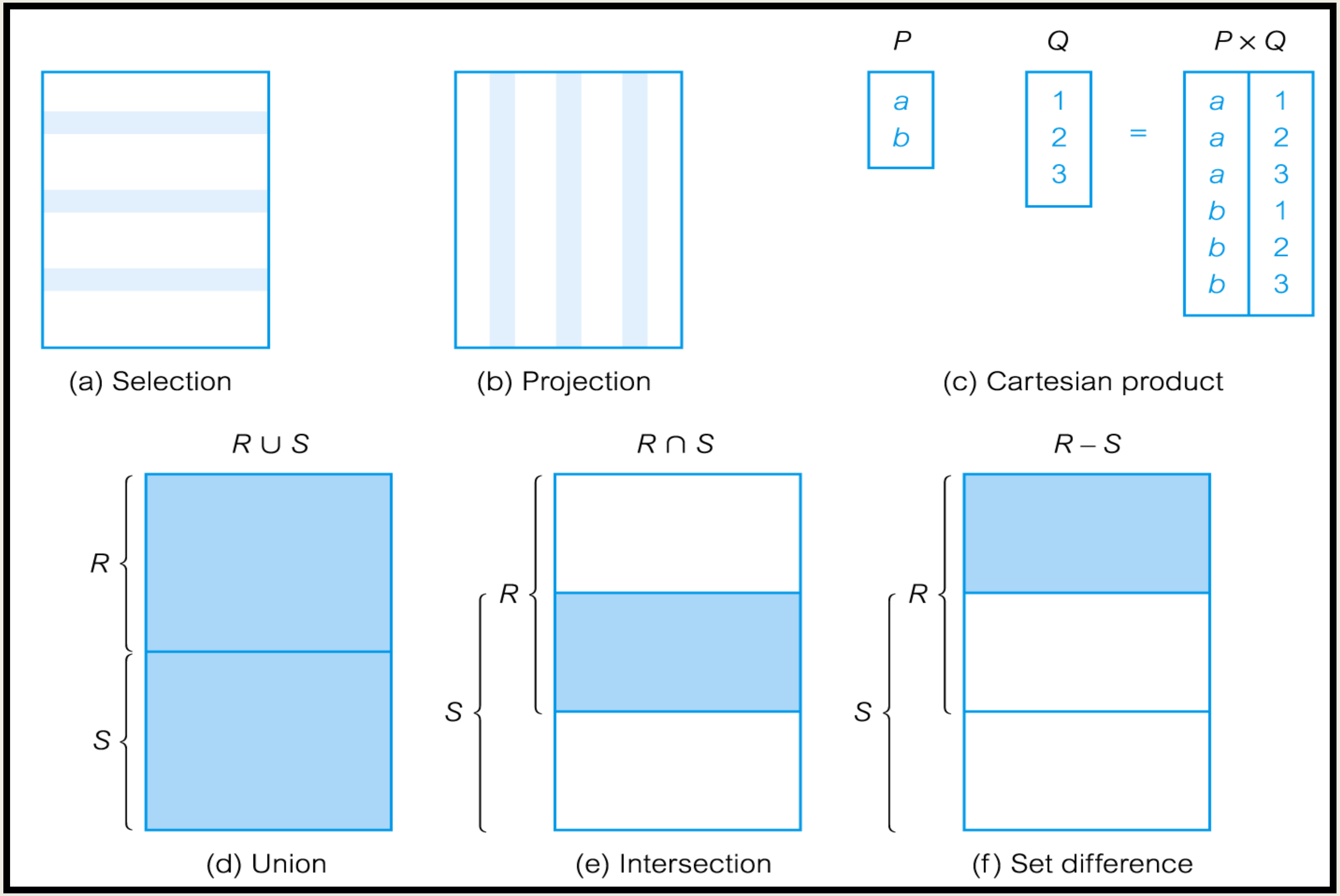
Unary Operation

- Selection
- Projection

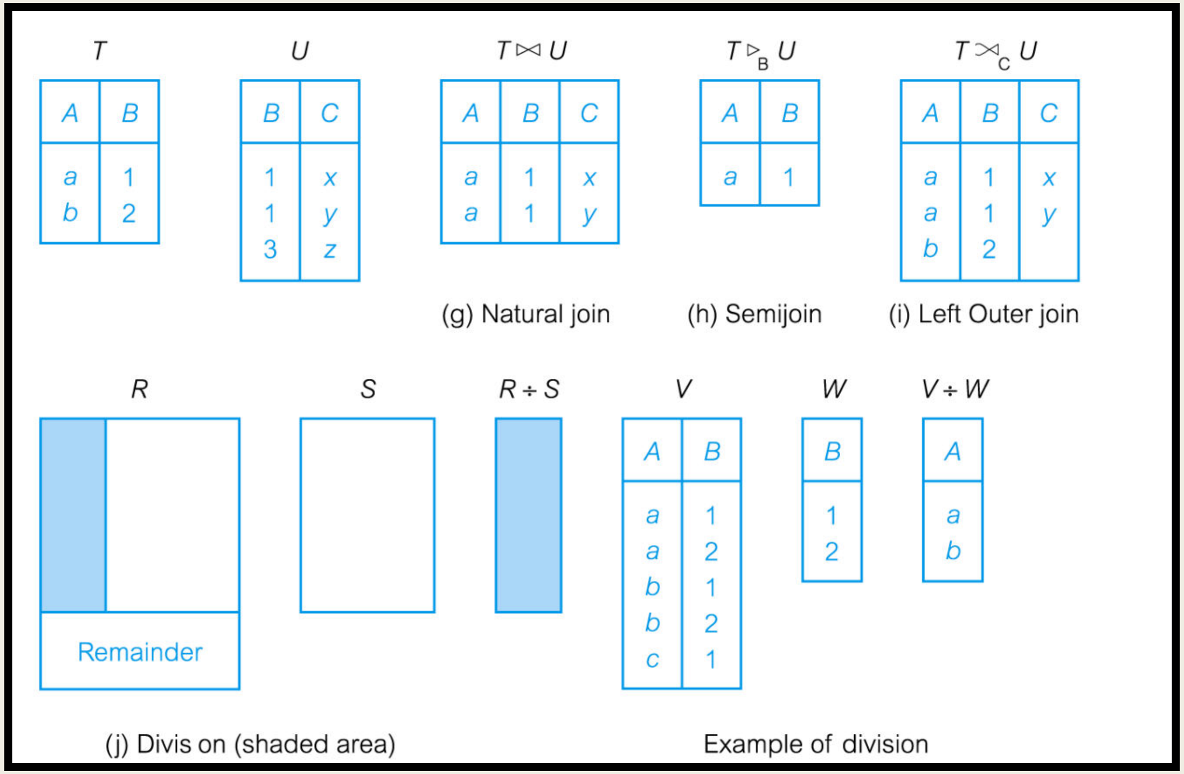
Binary Operation (Set Operation)

- Union
- Set Difference
- Cartesian Product
- Intersection
- Join
- Division

Relational Algebra Operations



Relational Algebra Operations



(g) Natural join (h) Semijoin (i) Left Outer join

(j) Divis on (shaded area) Example of division

Selection (Restriction)

- $\sigma_{predicate}$ (R)
  - *The selection operation works on a single relation R and defines a relation that contains only those tuples of R that satisfy the specified condition (predicate)*

Staff Number	Staff Name	Address

Selection (Restriction) - Example

- Example:

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

- List all staff with a salary greater than £10,000.

- Answer: $\sigma_{salary > 10000}$ (Staff)

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24- Mar-58	18000	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003

Selection (Restriction)

- More complex predicates can be generated using the logical operator
  - ^ *(AND),*
  - v *(OR)*
  - ~ *(NOT)*
- **Example:**

A	B	C
a	b	c
c	b	d

$\sigma_{B=b}(R)$

A	B	C
d	a	f

$\sigma_{\neg B=b}(R)$

A	B	C
a	b	c
c	b	d

$\sigma_{(B=b)\vee(A=a)}(R)$

A	B	C

$\sigma_{(B=b)\wedge(A=C)}(R)$

Projection

- $\Pi_{col1, \, . \, . \, . \, , \, coln}(R)$
  - *Works on a single relation R and defines a relation that contains a vertical subset of R, extracting the values of specified attributes and eliminating duplicates.*

Staff Number	Staff Name	Address

Projection - Example

- Example:
  - *Produce a list of salaries for all staff, showing only staffNo, fName, lName, and salary details.*

– *Answer:* $\Pi_{staffNo,\ fName,\ lName,\ salary}(Staff)$

staffNo	fName	lName	salary
SL21	John	White	30000
SG37	Ann	Beech	12000
SG14	David	Ford	18000
SA9	Mary	Howe	9000
SG5	Susan	Brand	24000
SL41	Julie	Lee	9000

Set Operations

- For purpose of combining more than one relations.
  - *Union*
  - *Set difference*
  - *Intersection*
  - *Cartesian product*

Union

- $R \cup S$
  - *Union of two relations R and S defines a relation that contains all the tuples of R, or S, or both R and S, duplicate tuples being eliminated.*
  - *R and S must be union-compatible.*

- Union compatible
  - *Union is possible only if the schemas of the two relations match, that is, if they have the same number of attributes with each pair of corresponding attributes having the same domain.*

- 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.

Union - Example

- List all cities where there is either a branch office or a property for rent.
  - $\Pi_{city}(Branch) \cup \Pi_{city}(PropertyForRent)$

branchNO	street	city	postcode

$\cup$

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo

city
London
Aberdeen
Glasgow
Bristol

Set Difference

- R – S
  - *The set difference operation defines a relation consisting of the tuples that are in relation R, but not in S. R and S must be union compatible.*

Set Difference - Example

- List all cities where there is a branch office but no properties for rent.
  - $\Pi_{city}(Branch) - \Pi_{city}(PropertyForRent)$

branchNO	street	city	postcode

—

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo

city
Bristol

Intersection

- $R \cap S$
  - *Defines a relation consisting of the set of all tuples that are in both R and S.*
  - *R and S must be union-compatible.*

- Can be expressed using basic operations:
  - $R \cap S = R - (R - S)$

Intersection - Example

List all cities where there is both a branch office and at least one property for rent.

$$\Pi_{city}(\text{Branch}) \cap \Pi_{city}(\text{PropertyForRent})$$

city
Aberdeen
London
Glasgow

Cartesian Product

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

- Can be expressed using basic operations:

  $R \cap S = R - (R - S)$

Cartesian Product - Example

List the names and comments of all clients who have viewed a property for rent.

$(\Pi_{clientNo,\ fName,\ lName}(Client))\ X\ (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing))$

client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR56	PA14	too small
CR76	John	Kay	CR76	PG4	too remote
CR76	John	Kay	CR56	PG4	
CR76	John	Kay	CR62	PA14	no dining room
CR76	John	Kay	CR56	PG36	
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR62	PA14	no dining room
CR56	Aline	Stewart	CR56	PG36	
CR74	Mike	Ritchie	CR56	PA14	too small
CR74	Mike	Ritchie	CR76	PG4	too remote
CR74	Mike	Ritchie	CR56	PG4	
CR74	Mike	Ritchie	CR62	PA14	no dining room
CR74	Mike	Ritchie	CR56	PG36	
CR62	Mary	Tregear	CR56	PA14	too small
CR62	Mary	Tregear	CR76	PG4	too remote
CR62	Mary	Tregear	CR56	PG4	
CR62	Mary	Tregear	CR62	PA14	no dining room
CR62	Mary	Tregear	CR56	PG36	

Cartesian Product - Example

- Use selection operation to extract those tuples where Client.clientNo = Viewing.clientNo.

$$\sigma_{Client.clientNo\ =\ Viewing.clientNo}((\Pi_{clientNo,\ fName,\ lName}(Client))\ X\ (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing)))$$

client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR56	PG36	
CR62	Mary	Tregear	CR62	PA14	no dining room

- Cartesian product and Selection can be reduced to a single operation called a Join.

Decomposing Complex Operations

- We can decompose RA operations into a series of smaller RA operations and give a name to the results of intermediate expressions.
- We can also use the Rename operations ρ (rho) which gives a name to the result of a RA operation.
- Rename allows an optional name for each of the attributes of the new relation to be specified.

$$\rho_S(E)$$

or

$$\rho_{S\ (a1,\ a2, \ldots an)}\ (E)$$

Join Operations

- Instead of a long Cartesian product operation we've used in previous example, we would normally use a Join operation.
- Join operation combines two relations to form a new relation
- Join is a derivative of Cartesian product, equivalent to performing a Selection, using join predicate as selection formula, over Cartesian product of the two operand relations.
- Join operations:

Theta Join

Equijoin

Natural Join

Outer Join

Semijoin

Theta Join (θ-join)

- $R \bowtie_F S$
  - *Defines a relation that contains tuples satisfying the predicate F from the Cartesian product of R and S*
  - *The predicate F is of the form $R.a_i \ \theta \ S.b_i$ where θ may be one of the comparison operators ($<, \leq, >, \geq, =, \neq$).*
  - *Theta join is actually equivalent to equation using Selection and Cartesian Product in the previous example:*

$$R \bowtie_F S = \sigma_F(R \times S)$$

  - *Degree of a Theta join is sum of degrees of the operand relations R and S. (Total number of attributes of relations R and S = Total number of attributes in a relation produced by Theta Join)*
  - *If predicate F contains only equality (=), the term Equijoin is used.*

Equijoin- Example

List the names and comments of all clients who have viewed a property for rent.

$$(\Pi_{clientNo,\ fName,\ lName}(Client)) \bowtie_{Client.clientNo\ =\ Viewing.clientNo} (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing))$$

client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR56	PG36	
CR62	Mary	Tregear	CR62	PA14	no dining room

Natural Join

- $R \bowtie S$
  - *An Equijoin of the two relations R and S over all common attributes x.*
  - *One occurrence of each common attribute is eliminated from the result.*

Natural Join - Example

List the names and comments of all clients who have viewed a property for rent.

$$(\Pi_{clientNo,\ fName,\ lName}(Client)) \bowtie (\Pi_{clientNo,\ propertyNo,\ comment}(Viewing))$$

clientNo	fName	lName	propertyNo	comment
CR76	John	Kay	PG4	too remote
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR56	Aline	Stewart	PG36	
CR62	Mary	Tregear	PA14	no dining room

Outer Join

- R ⟕ S
  - *We use Outer join when we want to display result of one relation that does not have matching tuple in the other relation (No matching values in the join attributes).*
  - *The (left) Outer join is a join in which tuples from R that do not have matching values in the common attributes of S are also included in the result relation.*
  - *Missing values in the second relation are set to null.*
  - *Outer join preserves tuples that would have been lost by other types of join.*
  - *Types of outer join: Left (natural) outer join, Right Outer join and Full Outer join*

Left Outer Join - Example

Produce a status report on property viewings.

$$\Pi_{propertyNo,\ street,\ city}(PropertyForRent) ⟕ Viewing$$

propertyNo	street	city	clientNo	viewDate	comment
PA14	16 Holhead	Aberdeen	CR56	24-May-01	too small
PA14	16 Holhead	Aberdeen	CR62	14-May-01	no dining room
PL94	6 Argyll St	London	null	null	null
PG4	6 Lawrence St	Glasgow	CR76	20-Apr-01	too remote
PG4	6 Lawrence St	Glasgow	CR56	26-May-01	
PG36	2 Manor Rd	Glasgow	CR56	28-Apr-01	
PG21	18 Dale Rd	Glasgow	null	null	null
PG16	5 Novar Dr	Glasgow	null	null	null

Semijoin

- $R \rhd_F S$
  - *Defines a relation that contains the tuples of R that participate in the join of R with S satisfying the predicate F.*
  - *Can rewrite Semijoin using Projection and Join:*

$$R \rhd_F S = \Pi_A(R \bowtie_F S)$$

Semijoin - Example

List complete details of all staff who work at the branch in Glasgow.

$$Staff \triangleright_{Staff.branchNo=Branch.branchNo}(\sigma_{city='Glasgow'}(Branch))$$

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24- Mar-58	18000	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003

Division Operation

- $R \div S$
  - *Defines a relation over the attributes C that consists of set of tuples from R that match combination of every tuple in S.*
  - *Assume that relation R is defined over the attribute set of A, relation S is defined over the attribute set B such that B is a subset of A. Let C = A – B where C is the set of attributes of R that are not attributes of S.*
  - *Expressed using basic operations:*

$$T_1 \leftarrow \Pi_C(R)$$

$$T_2 \leftarrow \Pi_C((S \times T_1) - R)$$

$$T \leftarrow T_1 - T_2$$

Division - Example

Identify all clients who have viewed all properties with three rooms.

$$(\Pi_{clientNo,\ propertyNo}(Viewing)) \div$$
$$(\Pi_{propertyNo}(\sigma_{rooms\ =\ 3}\ (PropertyForRent)))$$

$\Pi_{clientNo, propertyNo}(Viewing)$

clientNo	propertyNo
CR56	PA14
CR76	PG4
CR56	PG4
CR62	PA14
CR56	PG36

$\Pi_{propertyNo}(\sigma_{rooms=3}(PropertyForRent))$

propertyNo
PG4
PG36

RESULT

clientNo
CR56

Aggregate Operations

- $\mathfrak{I}_{AL}(R)$
  - *Applies aggregate function list, AL, to R to define a relation over the aggregate list.*
  - *AL contains one or more (<aggregate_function>, <attribute>) pairs .*
- Main aggregate functions are:

COUNT | SUM | AVG | MIN | MAX

Aggregate Operations - Example

How many properties cost more than £350 per month to rent?

$$\rho_R(myCount)\ \Im_{COUNT\ propertyNo}\ (\sigma_{rent > 350}\ (PropertyForRent))$$

myCount
5

(a)

Grouping Operations

- ${}_{GA}\mathfrak{I}_{AL}(R)$
  - *Groups tuples of R by grouping attributes, GA, and then applies aggregate function list, AL, to define a new relation.*
  - *AL contains one or more (<aggregate_function>, <attribute>) pairs.*
  - *Resulting relation contains the grouping attributes, GA, along with results of each of the aggregate functions.*

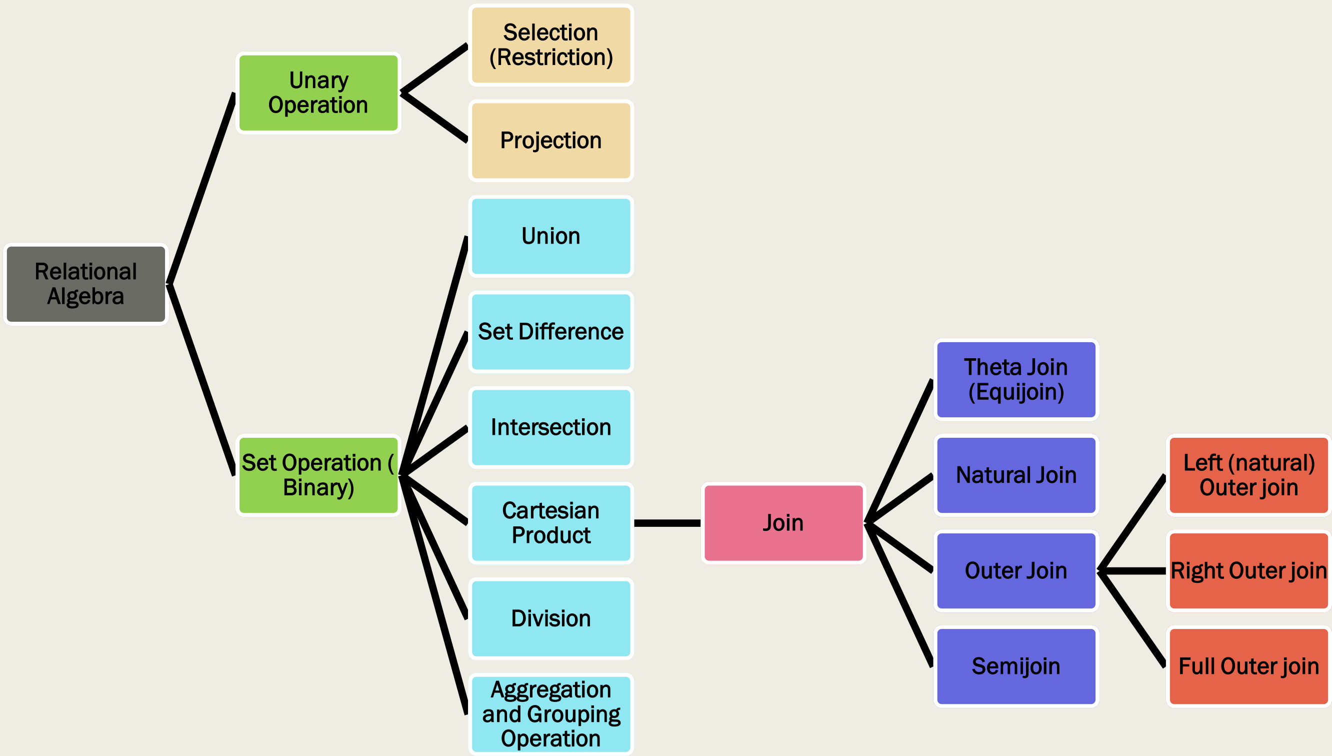
Grouping Operations - Example

Find the number of staff working in each branch and the sum of their salaries.

$$\rho_R(branchNo,\ myCount,\ mySum)\ _{branchNo}\,\mathfrak{I}_{COUNT\ staffNo,\ SUM\ salary}\ (Staff$$

branchNo	myCount	mySum
B003	3	54000
B005	2	39000
B007	1	9000

Summary


Relational Algebra
Unary Operation
Selection (Restriction)
Projection
Set Operation (Binary)
Union
Set Difference
Intersection
Cartesian Product
Division
Aggregation and Grouping Operation
Join
Theta Join (Equijoin)
Natural Join
Outer Join
Semijoin
Left (natural) Outer join
Right Outer join
Full Outer join

Reference

- *Database Systems: A Practical Approach to Design, Implementation, and Management,* Thomas Connolly and Carolyn Begg, 5th Edition, 2010, Pearson.

- Chapter 5