SECD2523 DATABASE

TOPIC 3 | RELATIONAL ALGEBRA

Content adapted from Connolly, T., Begg, C., 2015. Database Systems: A Practical Approach to Design, Implementation, and Management, Global Edition. Pearson Education.

LECTURE LEARNING OUTCOME

By the end of this lecture, students should be able to:

01 Identify relational algebra operations in relational algebra

02 Construct relational algebra statement to query relational database

INTRODUCTION

- Data model tells us the structure of data – requires way to query the data and to modify the data
  - We need to understand the operations done on relations
  - **RELATIONAL ALGEBRA** – a special algebra that consists of some simple but powerful ways to construct new relations from given relations
  - When the given relations are stored data, then the constructed relation can be answers to queries about this data

INTRODUCTION

- Algebra expression– consists of **operators** and **atomic operands**
  - In algebra of arithmetic, atomic operands are variables like ***x*** and constants like *15*
  - Operators are + - / * (examples)
  - An algebra expressions : ***(x + y) * z***

INTRODUCTION

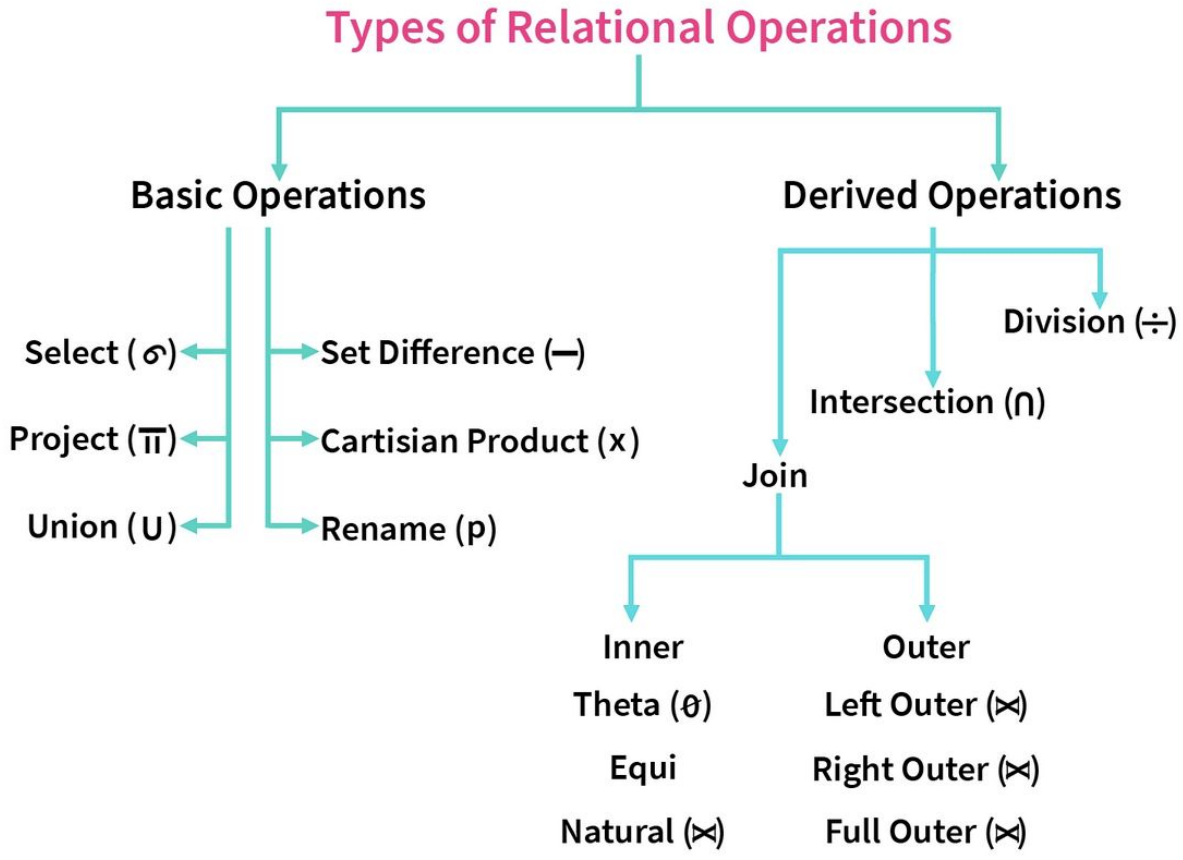
- Relational algebra – just like **normal algebra** that consists of **operators** and **atomic operands**
  - Atomic operands are:
    - Variables that stand for relations
    - Constants, which are finite relations
  - Operations classes:
    - Set operations
    - Operations that remove parts of a relation
    - Operations that combine tuples of 2 relations

INTRODUCTION

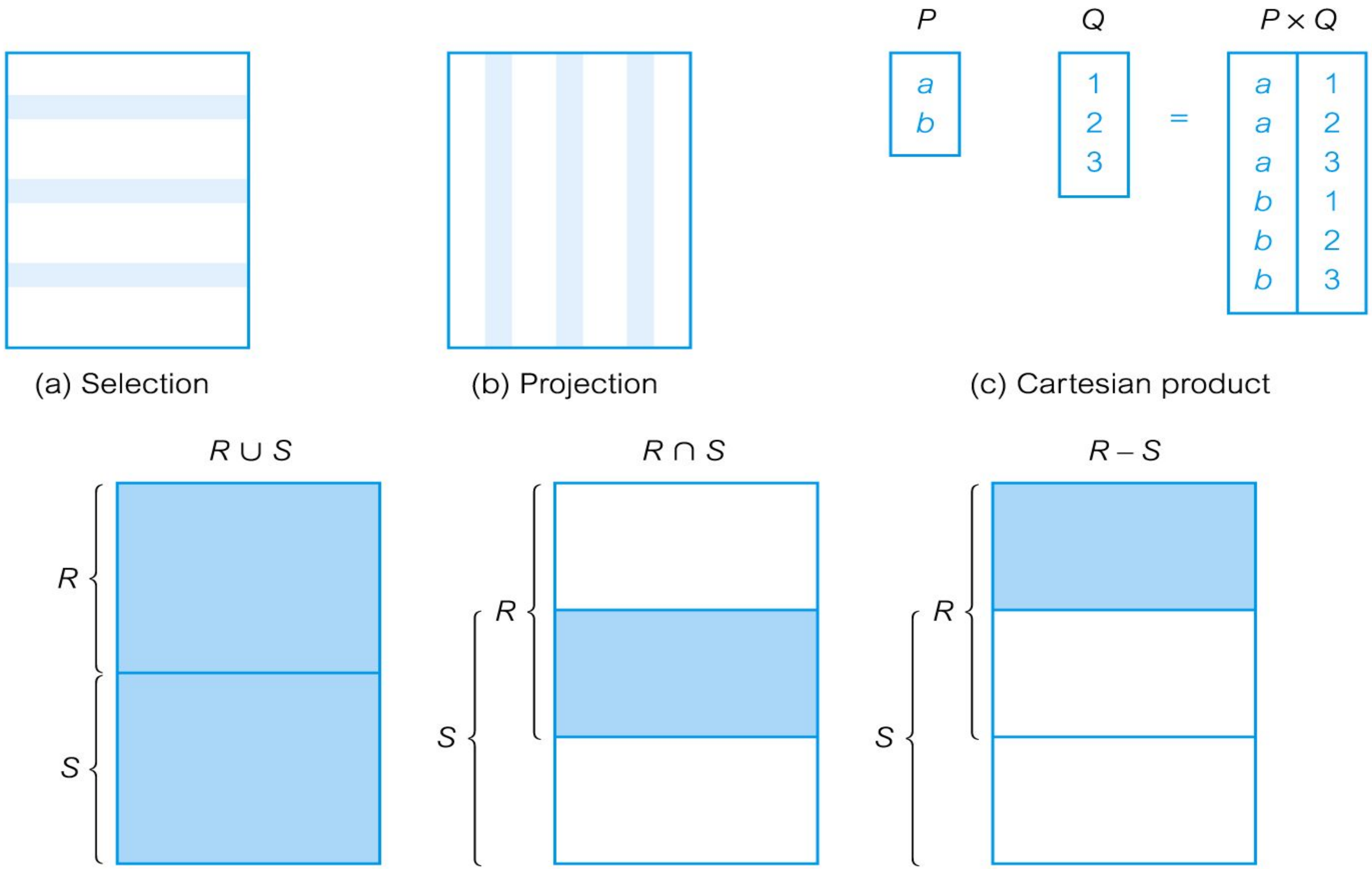
- Relational algebra operations work on one or more relations to define another relation without changing the original relations.
- Both operands and results are relations, so output from one operation can become input to another operation.
- Allows expressions to be nested (i.e. relations are closed under the algebra) - This property is called **closure**.

RELATIONAL ALGEBRA

- 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 also be expressed in terms of 5 basic operations.
- **Selection** and **Projection** are **unary** operations – operate on one relation
    - others are binary operations – requires pairs of relations



RELATIONAL ALGEBRA



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

RELATIONAL ALGEBRA

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

$T \triangleright_B U$

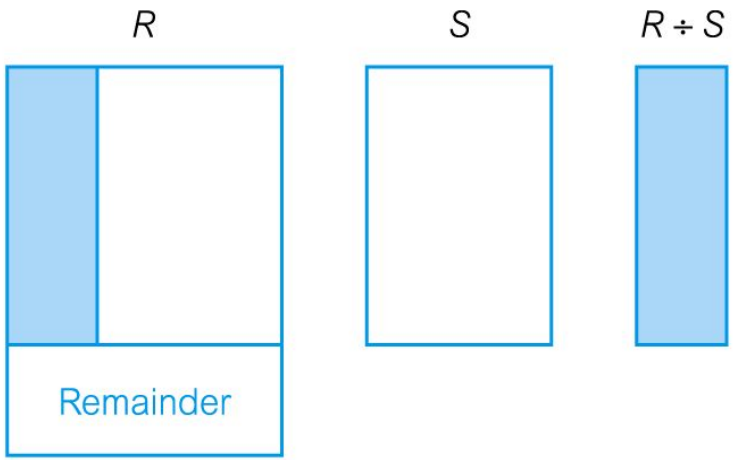
A	B
a	1

(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

We will use example in Database Systems 6th Ed. (Connolly & Begg, 2016).

Figure 4.3 from Chapter 4 The Relational Model (page 160)

Branch

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

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
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

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Client

clientNo	fName	lName	telNo	prefType	maxRent	eMail
CR76	John	Kay	0207-774-5632	Flat	425	john.kay@gmail.com
CR56	Aline	Stewart	0141-848-1825	Flat	350	astewart@hotmail.com
CR74	Mike	Ritchie	01475-392178	House	750	mritchie01@yahoo.co.uk
CR62	Mary	Tregear	01224-196720	Flat	600	maryt@hotmail.co.uk

PrivateOwner

ownerNo	fName	lName	address	telNo	eMail	password
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212	jkeogh@lhh.com	********
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419	cfarrel@gmail.com	********
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728	tinam@hotmail.com	********
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025	tony.shaw@ark.com	********

Viewing

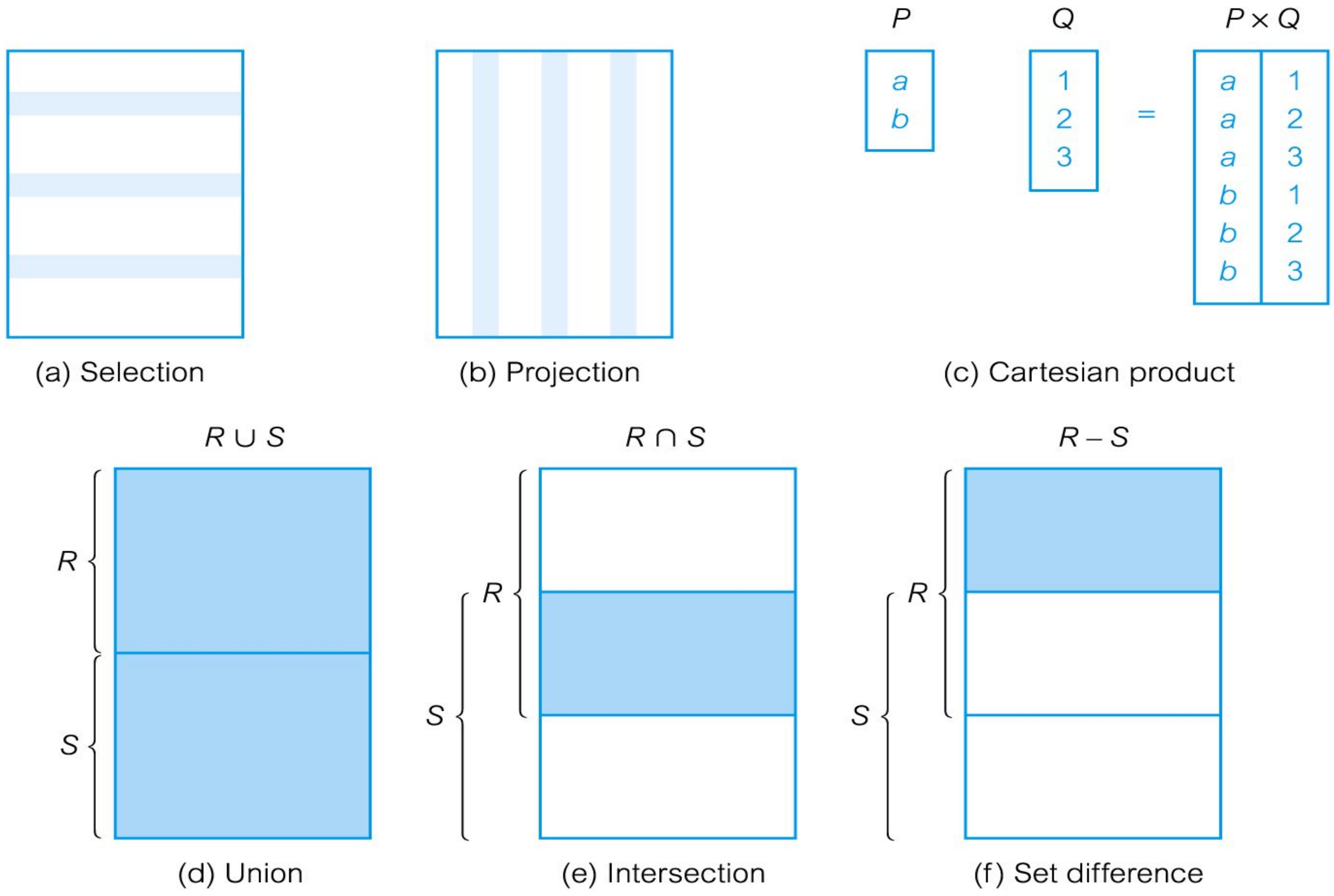
clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-13	too small
CR76	PG4	20-Apr-13	too remote
CR56	PG4	26-May-13	
CR62	PA14	14-May-13	no dining room
CR56	PG36	28-Apr-13	

Registration

clientNo	branchNo	staffNo	dateJoined
CR76	B005	SL41	2-Jan-13
CR56	B003	SG37	11-Apr-12
CR74	B003	SG37	16-Nov-11
CR62	B007	SA9	7-Mar-12

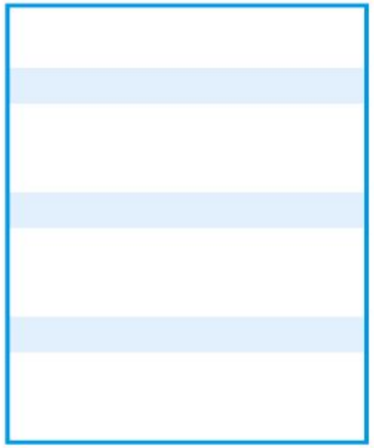
UTM UNIVERSITI TEKNOLOGI MALAYSIA

RELATIONAL ALGEBRA OPERATION I



SELECTION (σ)

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



- Works on a **single relation R**
- Defines a relation that contains **only those tuples (rows) of R** that **satisfy the specified condition (*predicate*).**

SELECTION (σ)

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

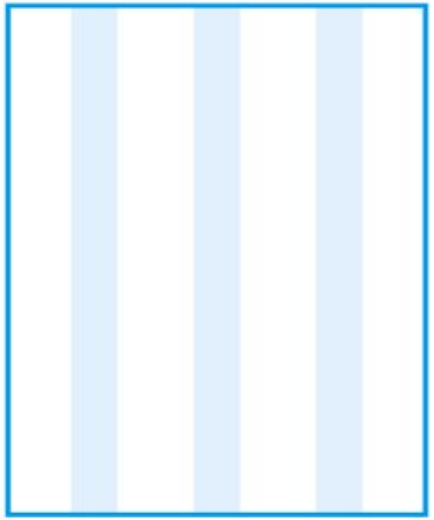
$$\sigma_{salary > 10000} \text{(Staff)}$$

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

UTM
UNIVERSITI TEKNOLOGI MALAYSIA

PROJECTION (Π)

$$\Pi col_1, \ldots, col_n(R)$$



- Works on a **single relation R**
- Defines a relation that contains **a vertical subset of R**
- Extracts the **values of specified attributes** and **eliminating duplicates**.

PROJECTION (Π)

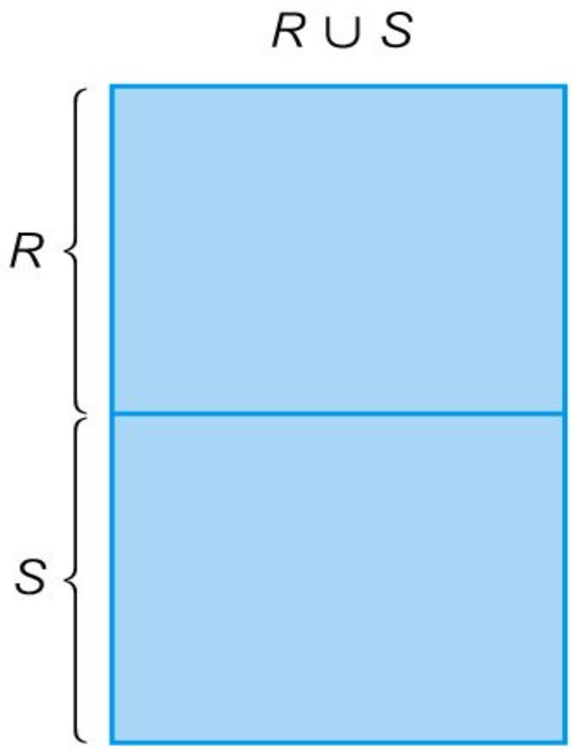
Produce a list of salaries for all staff, showing only staffNo, fName, lName, and salary details.

$$\Pi_{\text{staffNo, fName, lName, salary}}(\text{Staff})$$

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

UNION (U)

R U 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**.
  - **Degree of relation R and S is the same**
  - Domain for each pair of attribute in R and S is the same
    - R(A_1, A_2, A_3... A_n) ; S(B_1, B_2, B_3.... B_n) □ domain(A_i) = domain (B_i) for 1<=i<=n
- 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 (∪)

UTM UNIVERSITI TEKNOLOGI MALAYSIA

List all cities where there is either a branch office or a property for rent.

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

city
London
Aberdeen
Glasgow
Bristol

∪

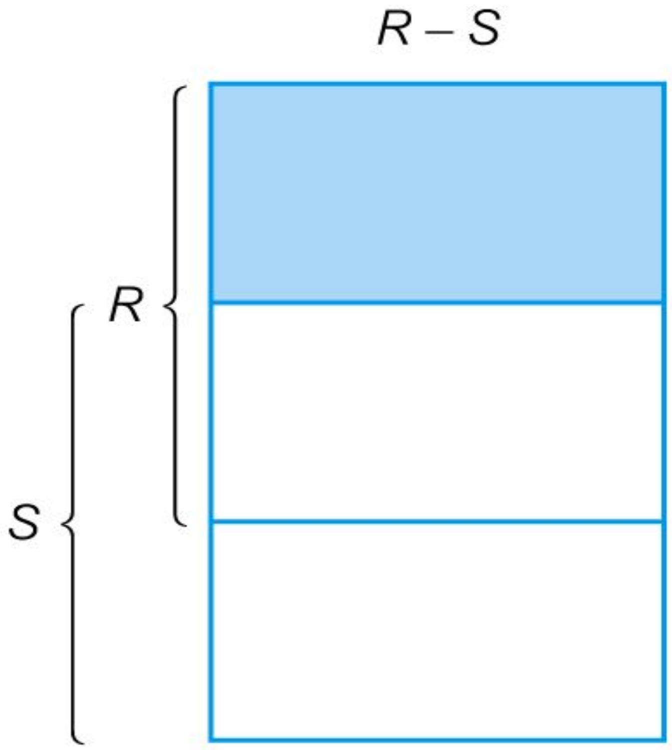
city
Aberdeen
London
Glasgow

=

city
London
Aberdeen
Glasgow
Bristol

SET DIFFERENCE (−)

R − S



- 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 (–)

List all cities where there is a branch office but no properties for rent.

$$\Pi_{city}(Branch) - \Pi_{city}(PropertyForRent)$$

city
London
Aberdeen
Glasgow
Bristol

\-

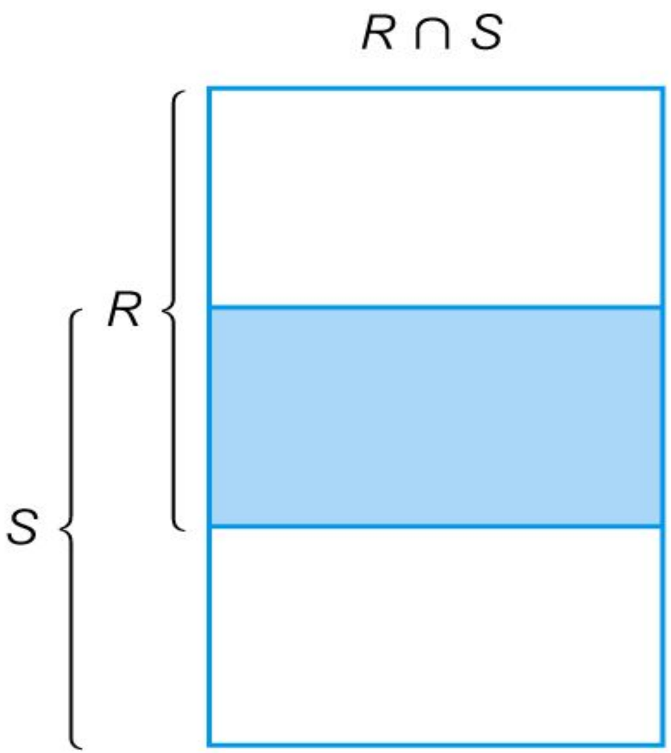
city
Aberdeen
London
Glasgow

=

city
Bristol

INTERSECTION (∩)

R ∩ 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**.
- Expressed using basic operations:
  R ∩ S = R – (R – S)

INTERSECTION (∩)

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
London
Aberdeen
Glasgow
Bristol

∩

city
Aberdeen
London
Glasgow

=

city
Aberdeen
London
Glasgow

CARTESIAN PRODUCT (X)

R X S

P

a
b

Q

1
2
3

=

$P \times Q$

a	1
a	2
a	3
b	1
b	2
b	3

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

CARTESIAN PRODUCT (X)

UTM UNIVERSITI TEKNOLOGI MALAYSIA

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

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

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

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	

2. Use **Cartesian Product** (X) and **Project** selected column

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***.

CARTESIAN PRODUCT (X)

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

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*.

CLASS DISCUSSION I

Describe the result of the relational algebra expression:

1. $\Pi_{hotelNo}(\sigma_{price > 50}(Room))$

2. $\sigma_{Hotel.hotelNo = Room.hotelNo}(Hotel \times Room)$

CLASS DISCUSSION I

Write the relational algebra expression for the following statements:

1. List all hotels.
2. List all single rooms with a price below RM200 per night.
3. List the names and cities of all guests.

RELATIONAL ALGEBRA II

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

JOIN OPERATIONS

- 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.
- One of the most difficult operations to implement efficiently in an RDBMS and one reason why RDBMSs have intrinsic performance problems.
- Various forms of join operation
  - Theta join
  - Equijoin (a particular type of Theta join)
  - 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 **(<, ≤, >, ≥, =, ≠)**.

THETA JOIN (θ-JOIN)

- Can rewrite Theta join using basic Selection and Cartesian product operations.

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

- Degree of a Theta join is sum of degrees of the operand relations R and S.

- Example:

List the names of clients with the apartment they can afford.

$$\text{TempClient} \bowtie_{\text{TempClient.maxRent} \leq \text{TempPropertyForRent.rent}} \text{TempPropertyForRent}$$

EQUIJOIN

- If predicate F contains only equality (=), the term ***Equijoin*** is used.
- Example:

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

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

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

$$R \bowtie S$$

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

- 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

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

$T ⟕_C U$

A	B	C
a	1	x
a	1	y
b	2	

- To display rows in the result that **do not have matching values** in the join column, use **OUTER JOIN**.
- **(Left) outer join** is a join in which **tuples from R that do not have matching values in common columns of S** are also included in result relation.

OUTER JOIN (⟕)

Produce a status report on property viewings.

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

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 (▷)

$$R \triangleright S$$

$T \triangleright_B U$

A	B
a	1

- Can rewrite Semijoin using Projection and Join:

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

- Defines a relation that contains the **tuples of R that participate in the join of R with S**.

SEMIJOIN (▷)

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

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

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

CLASS DISCUSSION II

Describe the result of the relational algebra expression based on the following schema:

The following tables form part of a database held in a relational DBMS:

Hotel (<u>hotelNo</u>, hotelName, city)
Room (<u>roomNo</u>, <u>hotelNo</u>, type, price)
Booking (<u>hotelNo</u>, <u>guestNo</u>, <u>dateFrom</u>, dateTo, roomNo)
Guest (<u>guestNo</u>, guestName, guestAddress)

where Hotel *contains hotel details and* hotelNo *is the primary key;*
Room contains room details for each hotel and (roomNo, hotelNo) forms the primary key;
Booking contains details of bookings and (hotelNo, guestNo, dateFrom) forms the primary key;
Guest contains guest details and guestNo is the primary key.

1. $\Pi_{hotelName}$ (Hotel $\bowtie_{Hotel.hotelNo = Room.hotelNo}$ ($\sigma_{price > 50}$ (Room)))
2. Guest $⟕$ ($\sigma_{dateTo \geq \text{‘1-Jan-2002’}}$ (Booking))
3. Hotel $\triangleright_{Hotel.hotelNo = Room.hotelNo}$ ($\sigma_{price > 50}$ (Room)))
4. $\Pi_{guestName, hotelNo}$ (Booking $\bowtie_{Booking.guestNo = Guest.guestNo}$ Guest) $\div$ $\Pi_{hotelNo}$ ($\sigma_{city = \text{’London’}}$(Hotel))

CLASS DISCUSSION II

2. Guest $⟕$ ($\sigma_{dateTo \geq \text{'1-Jan-2002'}}$ (Booking))

This will produce a **(left outer) join of Guest and those tuples of Booking** with an **end date (dateTo) greater than or equal to 1-Jan-2002**.

All guests who don't have a booking with such a date will still be included in the join.

Essentially this will produce a **relation containing all guests and show the details of any bookings they have beyond 1-Jan-2002**.

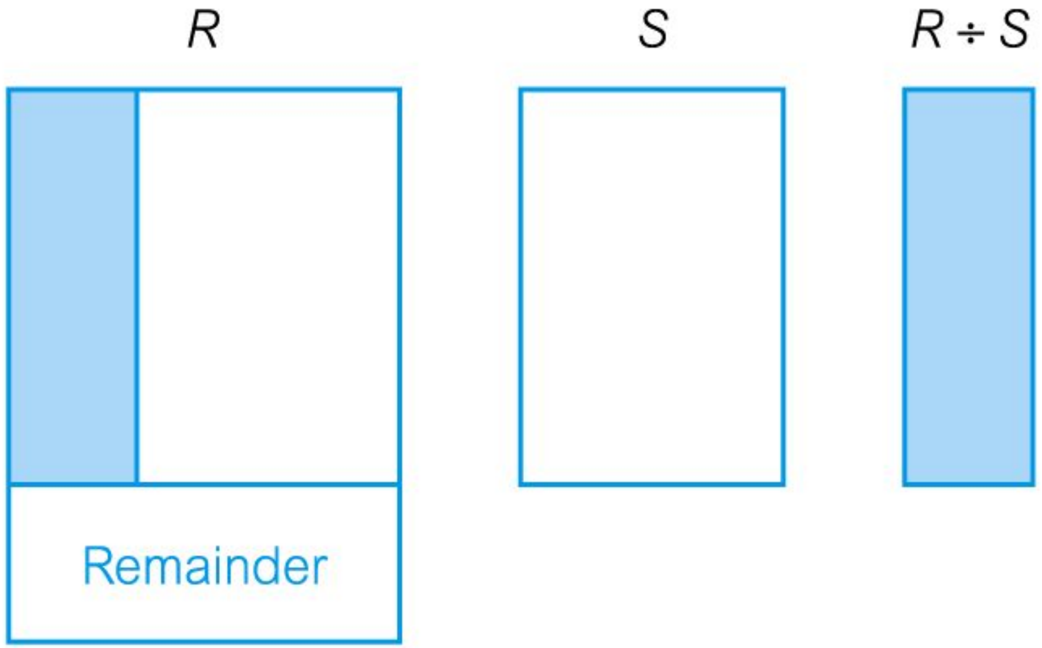
CLASS DISCUSSION II

UTM UNIVERSITI TEKNOLOGI MALAYSIA

Write the relational algebra expression for the following statements (refer to the Hotel schema in previous question):

1. List the price and type of all rooms at the Grosvenor Hotel.
2. List all guests currently staying at the Grosvenor Hotel.
3. List the details of all rooms at the Grosvenor Hotel, including the name of the guest staying in the room, if the room is occupied.
4. List the guest details (guestNo, guestName, and guestAddress) of all guests staying at the Grosvenor Hotel.

RELATIONAL ALGEBRA III



V

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

W

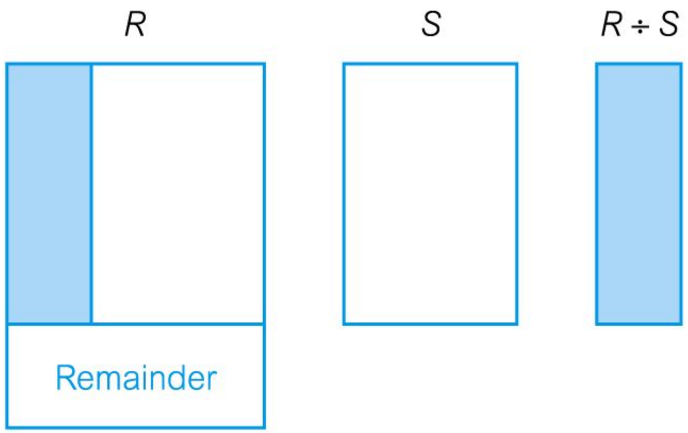
B
1
2

V ÷ W

A
a
b

DIVISION (÷)

$$R \div S$$



Defines a relation over the **attributes C that consists of set of tuples from R that match combination of *every* tuple in S**.

Expressed using basic operations:

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

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

$$T \leftarrow T_1 - T_2$$

V

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

W

B
1
2

V ÷ W

A
a
b

Example of division

DIVISION (÷)

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

$\Im AL(R)$

- 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**, and **MAX**.

AGGREGATE OPERATIONS

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

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

Rename operation ~rho

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

count

myCount
5

AGGREGATE OPERATIONS

$$GA\,\Im\,AL(R)$$

- 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.

AGGREGATE OPERATIONS

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

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

Staff

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
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007

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

CLASS DISCUSSION III

Describe the result of the relational algebra expression

1. $\Pi_{guestName,\ hotelNo}$ (Booking $\bowtie_{Booking.guestNo\ =\ Guest.guestNo}$ Guest) $\div$ $\Pi_{hotelNo}$ ($\sigma_{city\ =\ 'London'}$(Hotel))

SUMMARY

- The relational algebra is a (high-level) procedural language: it can be used to tell the DBMS how to build a new relation from one or more relations in the database.

- The five fundamental operations in relational algebra [**Selection, Projection, Cartesian product, Union, and Set difference**] perform most of the data retrieval operations that we are interested in.

- In addition, there are also the **Join**, **Intersection**, and **Division** operations, which can be expressed in terms of the five basic operations.

Innovating Solutions

Menginovasi Penyelesaian