IBA

Relational Algebra

CS 341 Database Systems

Introduction to Relational Algebra

- Form the basis for the widely used SQL query language

- Database systems do not allow users to write queries in relational algebra

- The relational algebra is very important for several reasons:
    1. It provides a **formal foundation** for relational model operations.
    2. And perhaps more important, it is used as a **basis for implementing and optimizing queries** in the query processing and optimization modules that are integral parts of relational database management systems (RDBMSs)
    3. **Some of its concepts** are incorporated into the SQL standard query language for RDBMSs.

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 called *closure.*

6 Basic Operations

- select: σ
- project: Π
- union: $\cup$
- set difference: $-$
- Cartesian product: x
- rename: ρ

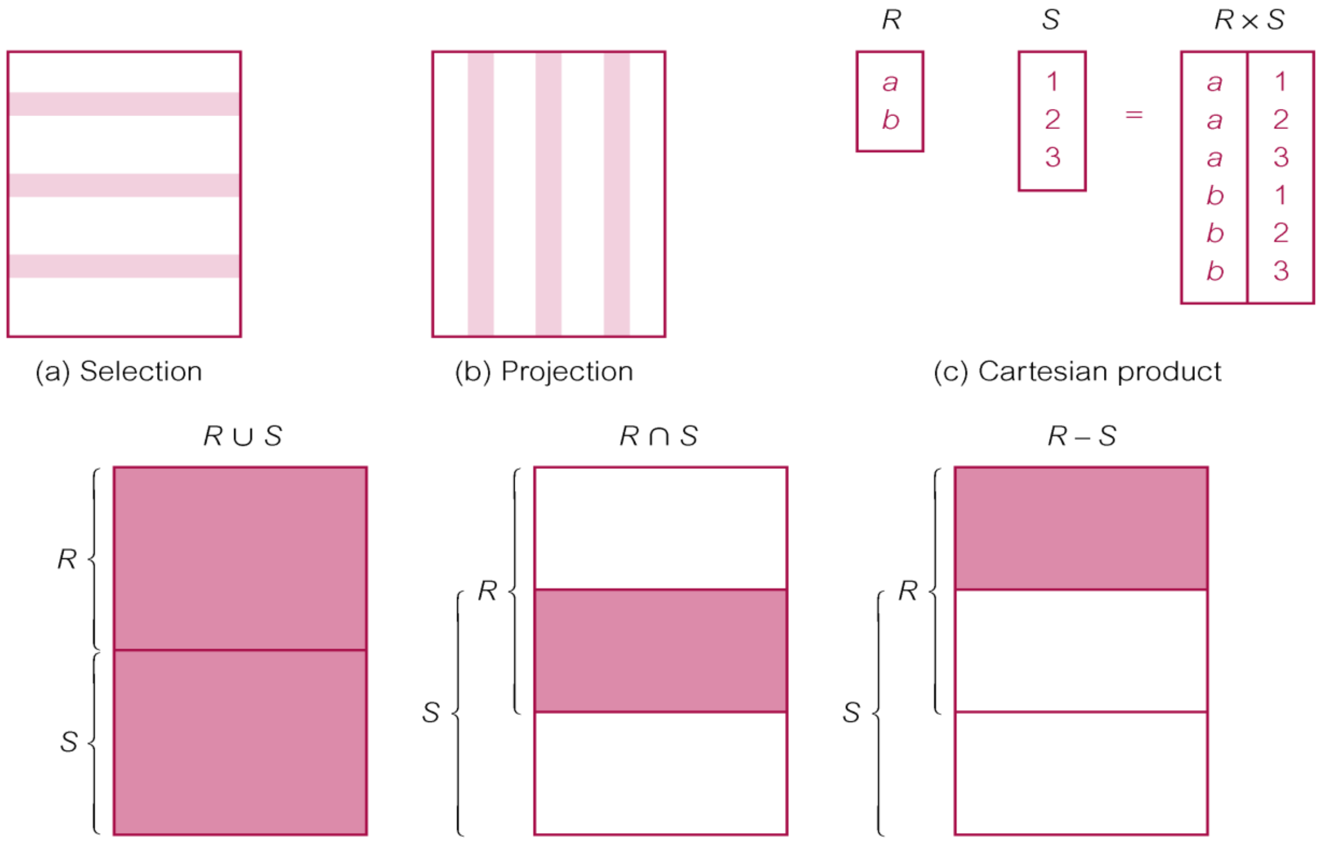
- The operators take one or two relations as inputs and produce a new relation as a result.
- **Unary operations**: select, project, and rename (operates on one relation)
- **Binary**: union, Cartesian product, and set difference (operates on pairs of relations)

Derived Operators

- Join: $\bowtie$
- Intersection: $\cap$
- Division: / or ÷

- Can be expressed in terms of 6 basic operations.

Relational Algebra Operations



(a) Selection

(b) Projection

(c) Cartesian product

(d) Union

(e) Intersection

(f) Set difference

Relational Algebra Operations

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

$T \triangleright_B U$

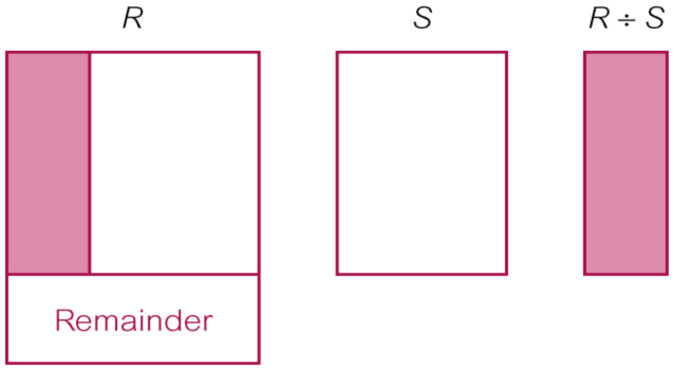
A	B
a	1

(h) Semijoin

$T ⟕_C U$

A	B	C
a	1	x
a	1	y
b	2	

(i) Left Outer join



(j) Division (shaded area)

V

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

W

B
1
2

$V \div W$

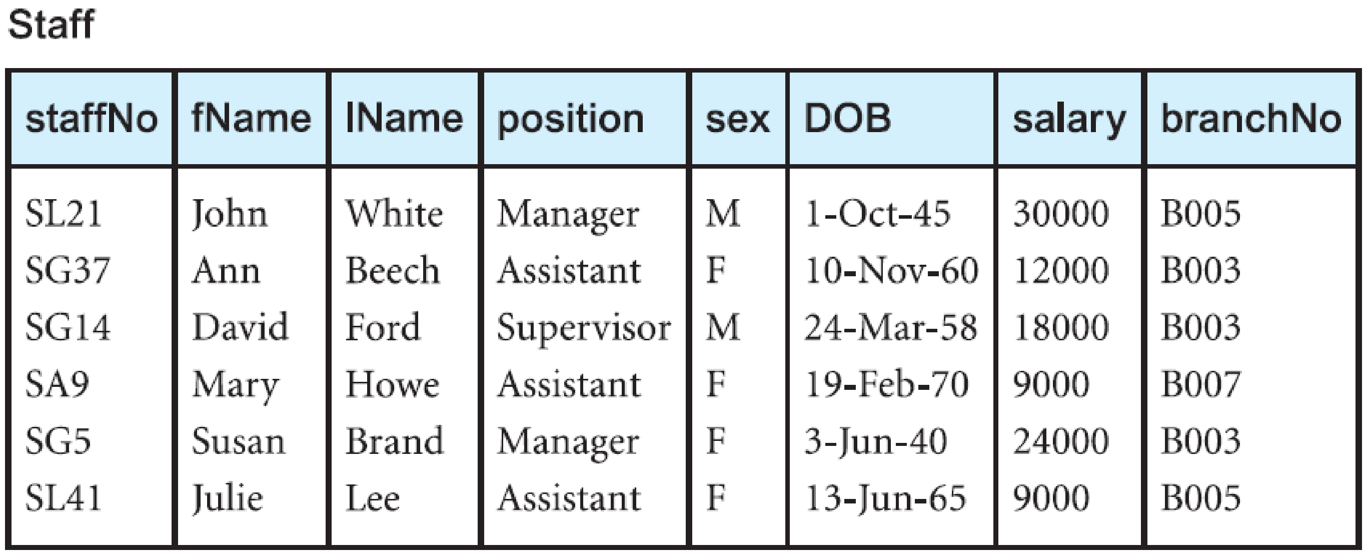
A
a
b

Example of division

Selection (or Restriction)

- **$\sigma_{predicate}$ (R)**
  - Works on a single relation R and defines a relation that contains only those tuples (rows) of R that satisfy the specified condition (predicate).

Staff Table



Staff

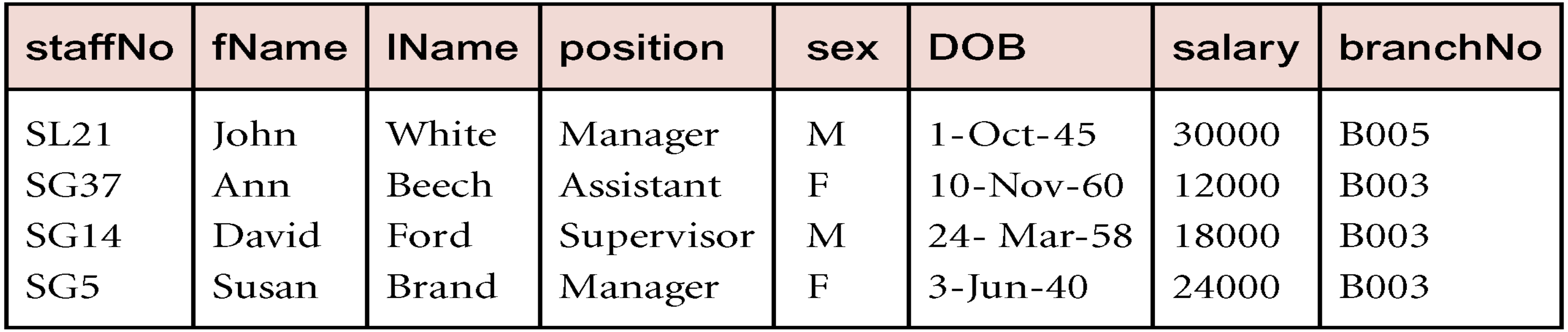
staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee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

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

Example - Selection (or Restriction)

- 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

Logical Operators

- When there are multiple conditions, they can be separated by logical operators
- Λ AND
- V OR
- ~ NOT

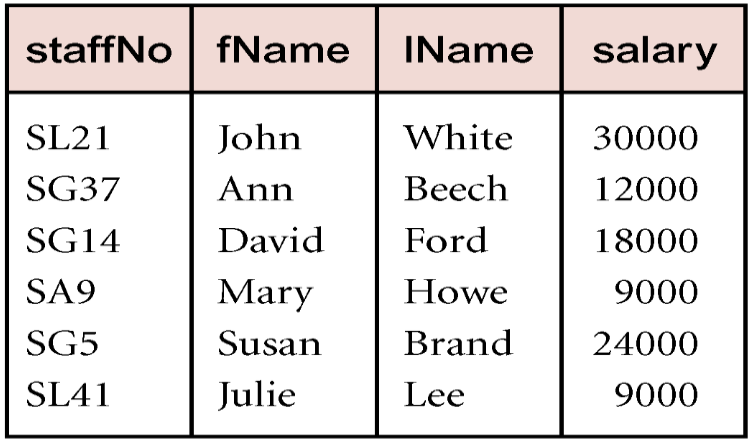
Projection

- $\Pi_{\text{col1}, \ldots, \text{coln}}(R)$
  - 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.*

Example - Projection

- 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

Select and Project

$\Pi_{\text{staffNo, fName, lName, salary}}(\sigma_{\text{salary > 10000}} (\text{Staff}))$

- What if we project first and then select?

  $\sigma_{\text{salary > 10000}} (\Pi_{\text{staffNo, fName, lName, salary}} (\text{Staff}))$

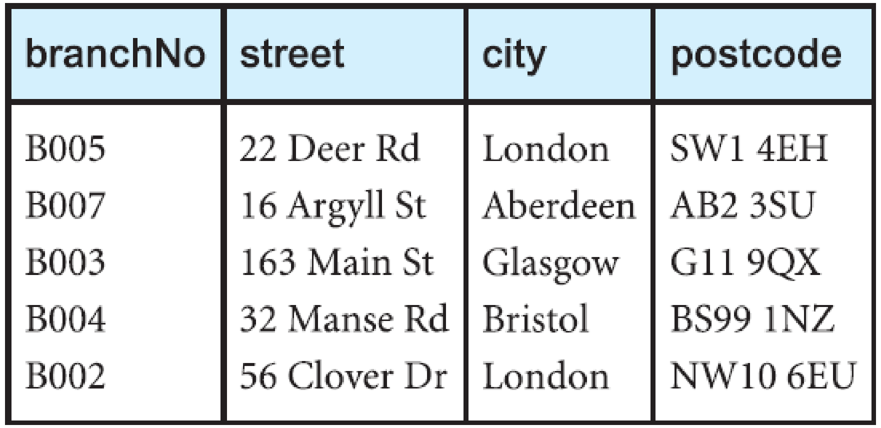
  $\sigma_{\text{salary > 10000}} (\Pi_{\text{staffNo, fName, lName}} (\text{Staff}))$

Union

- **R ∪ 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 property means: Both the relations must have same attribute characteristics i.e number of columns and their data types must match.*

- 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.

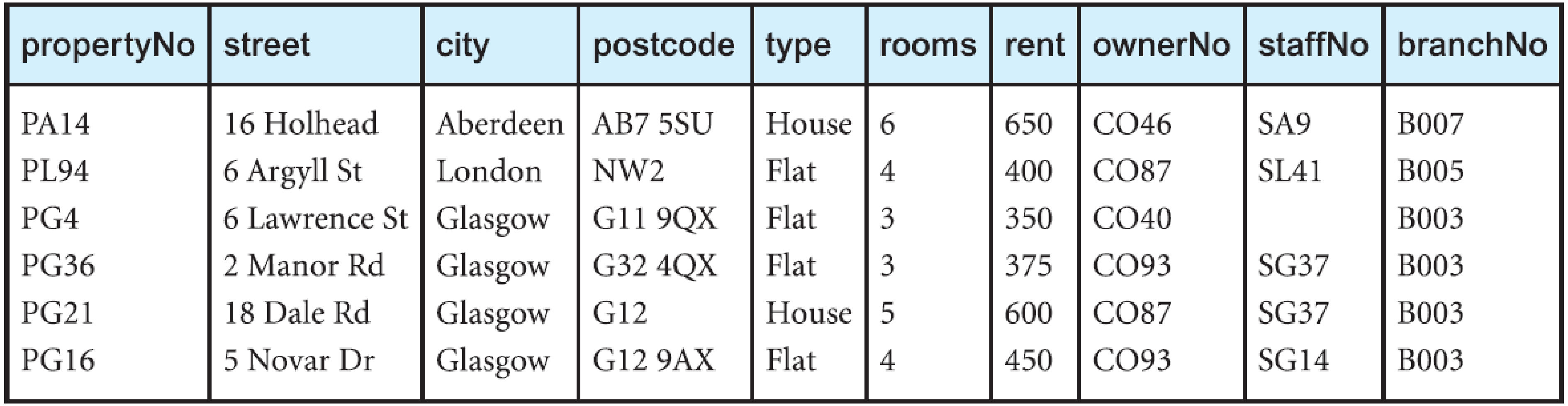
Branch



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

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

PropertyForRent

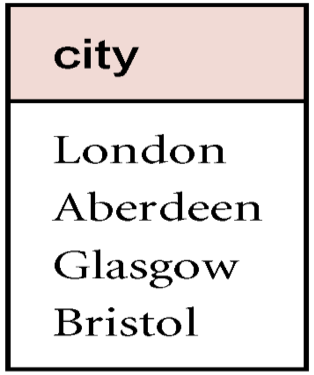


propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21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

Example - Union

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

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



city
London
Aberdeen
Glasgow
Bristol

Union is also used as a command in SQL.

The Assignment Operation

- The assignment operation is denoted by ← and works like assignment in a programming language.
- Example: Find all instructor in the “Physics” and Music department.

$$Physics \leftarrow \sigma_{dept\_name=\text{“Physics”}}(instructor)$$
$$Music \leftarrow \sigma_{dept\_name=\text{“Music”}}(instructor)$$

$$Physics \cup Music$$

- With the assignment operation, a query can be written as a sequential program consisting of a *series of assignments followed by an expression whose value is displayed as the result of the query.*

The Rename Operation

- The results of relational-algebra expressions do not have a name that we can use to refer to them. The **rename operator, ρ** is provided for that purpose
- The expression:

$$\rho_x (E)$$

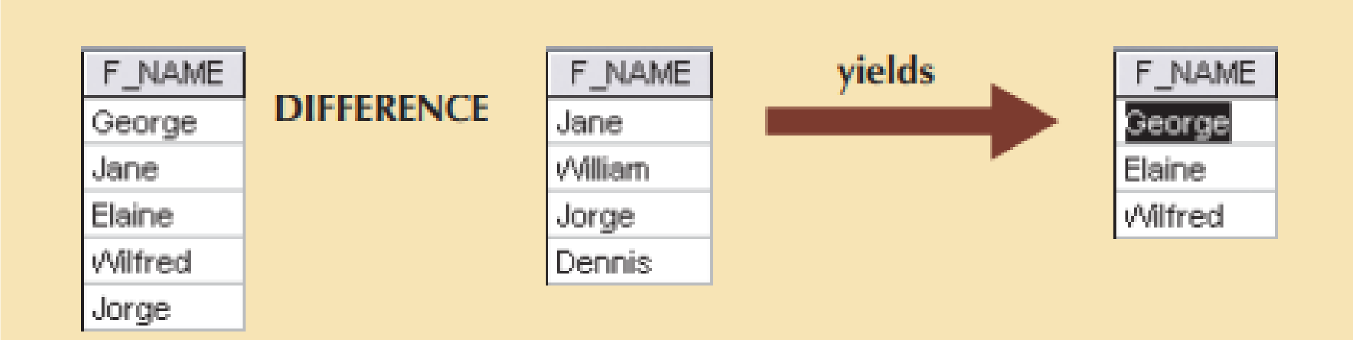
returns the result of expression *E* under the name *x*

- Another form of the rename operation: (also renames attributes)

$$\rho_{x(A1,A2, .. An)} (E)$$

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.



F_NAME
George
Jane
Elaine
Wilfred
Jorge

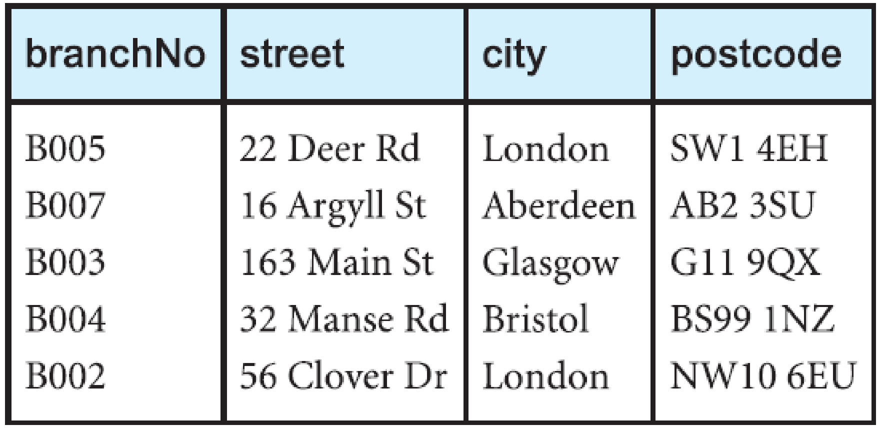
DIFFERENCE

F_NAME
Jane
William
Jorge
Dennis

yields

F_NAME
George
Elaine
Wilfred

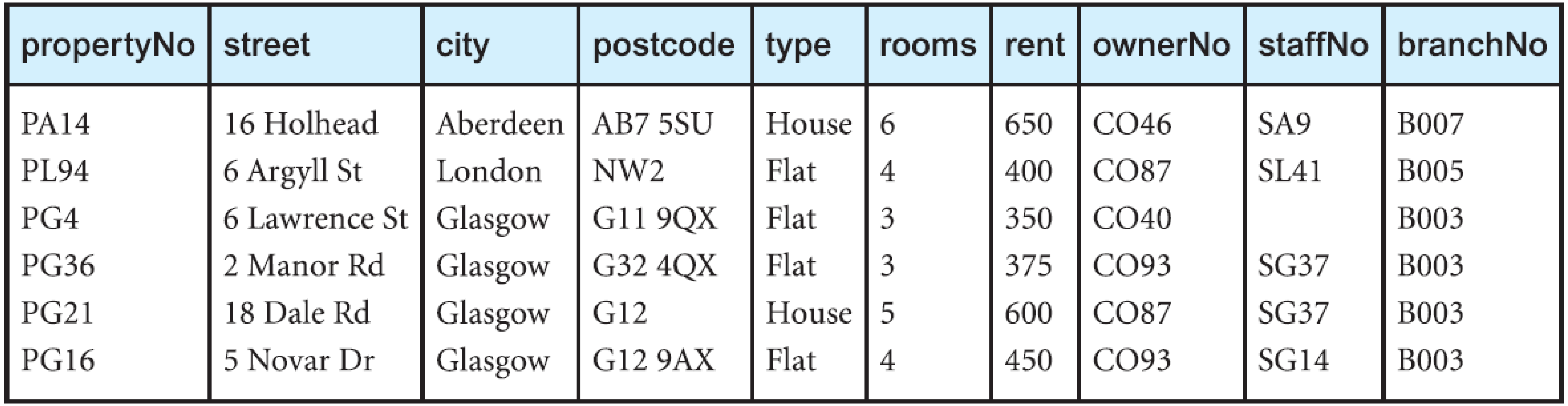
Branch



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

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

PropertyForRent

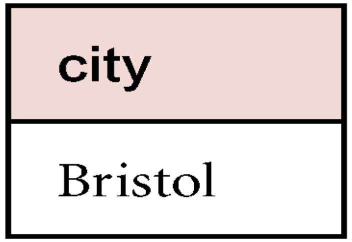


propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21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

Example - Set Difference

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

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



city
Bristol

Minus in SQL is same as Set difference.

Equivalent Queries

- There is more than one way to write a query in relational algebra.
- Example: Find information about courses taught by instructors in the Physics department with salary greater than 90,000
- Query 1

$$\sigma_{\textit{dept\_name}=\text{“}\textit{Physics}\text{”} \wedge \textit{salary} > 90{,}000}(\textit{instructor})$$

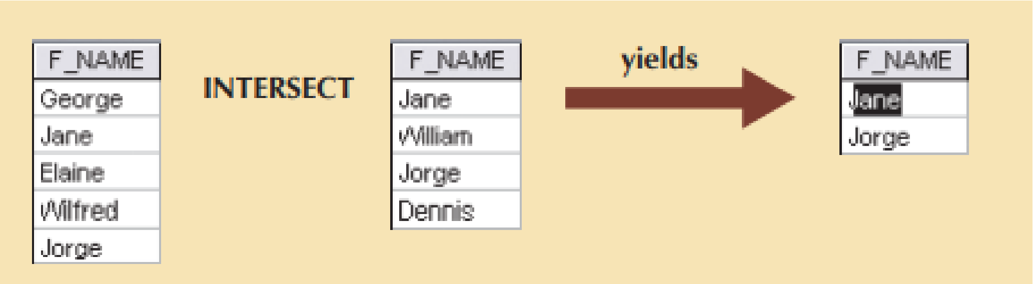
- Query 2

$$\sigma_{\textit{dept\_name}=\text{“}\textit{Physics}\text{”}}(\sigma_{\textit{salary} > 90{,}000}(\textit{instructor}))$$

- The two queries are not identical; they are, however, ***equivalent*** -- they give the same result on any database.

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.



F_NAME
George
Jane
Elaine
Wilfred
Jorge

INTERSECT

F_NAME
Jane
William
Jorge
Dennis

yields

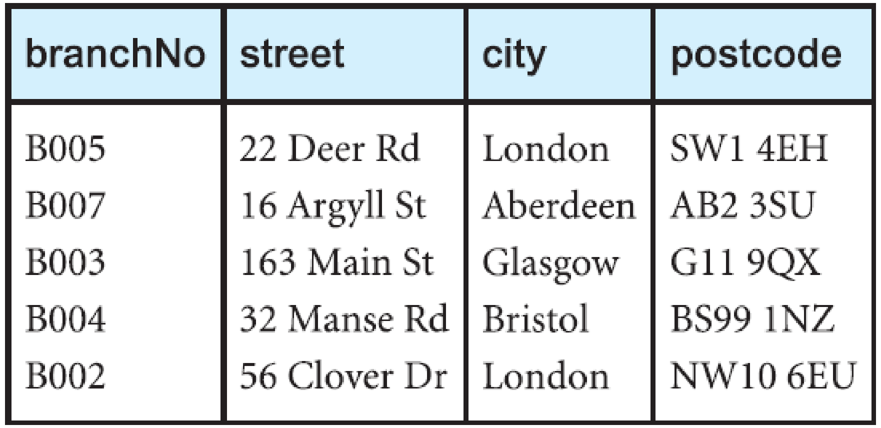
F_NAME
Jane
Jorge

- Expressed using basic operations:

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

Intersect is also used as a command in SQL.

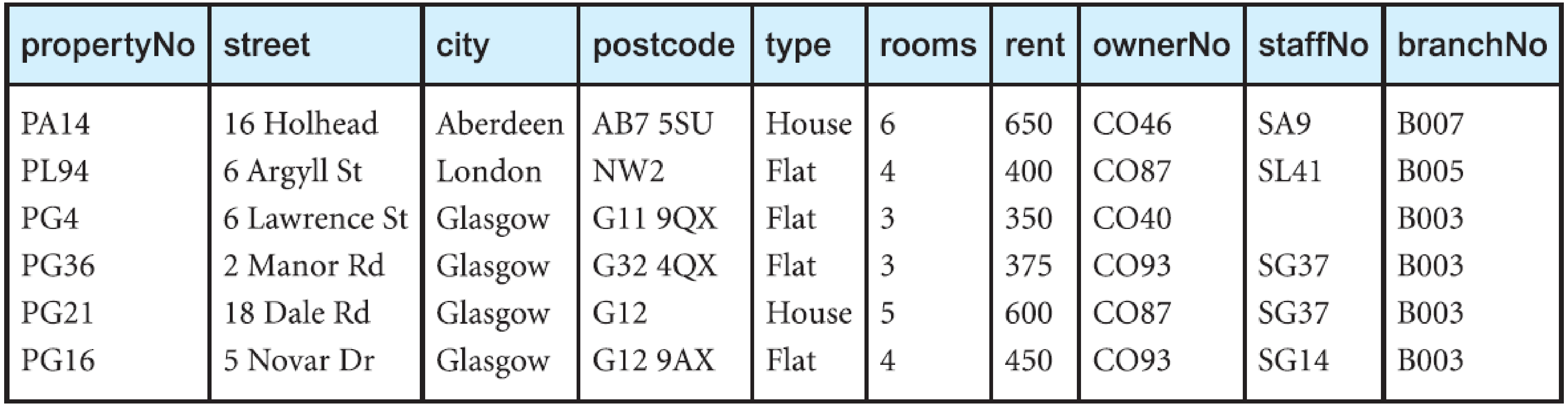
Branch



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

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

PropertyForRent

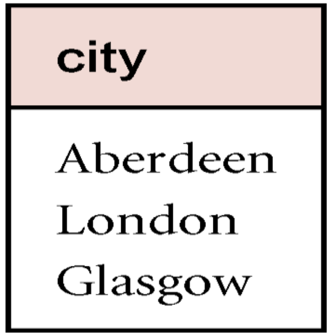


propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21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

Example - Intersection

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

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

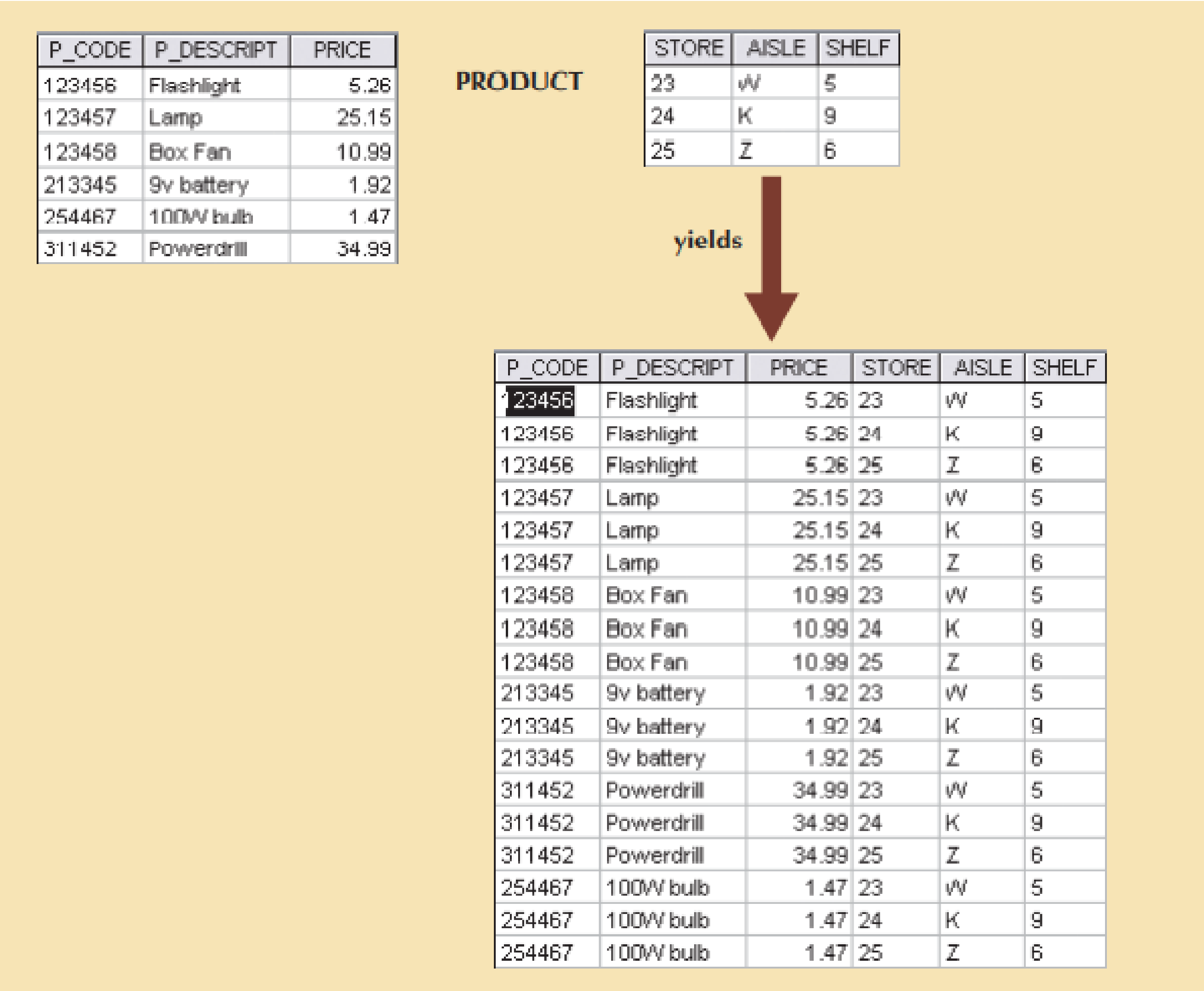


city
Aberdeen
London
Glasgow

Cartesian Product

- **R X S**
  - Defines a relation that is the concatenation of every tuple of relation R with every tuple of relation S.
  - *Yields all possible pairs of rows from two tables*
  - If one table has six rows and the other table has three rows, the PRODUCT yields a list composed of 6 × 3 = 18 rows.

Cartesian Product



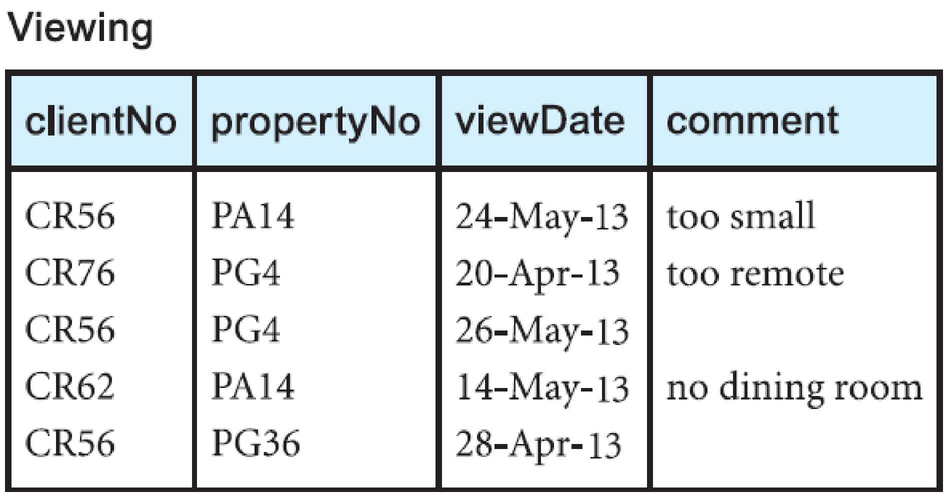
P_CODE	P_DESCRIPT	PRICE
123456	Flashlight	5.26
123457	Lamp	25.15
123458	Box Fan	10.99
213345	9v battery	1.92
254467	100W bulb	1.47
311452	Powerdrill	34.99

PRODUCT

STORE	AISLE	SHELF
23	W	5
24	K	9
25	Z	6

yields

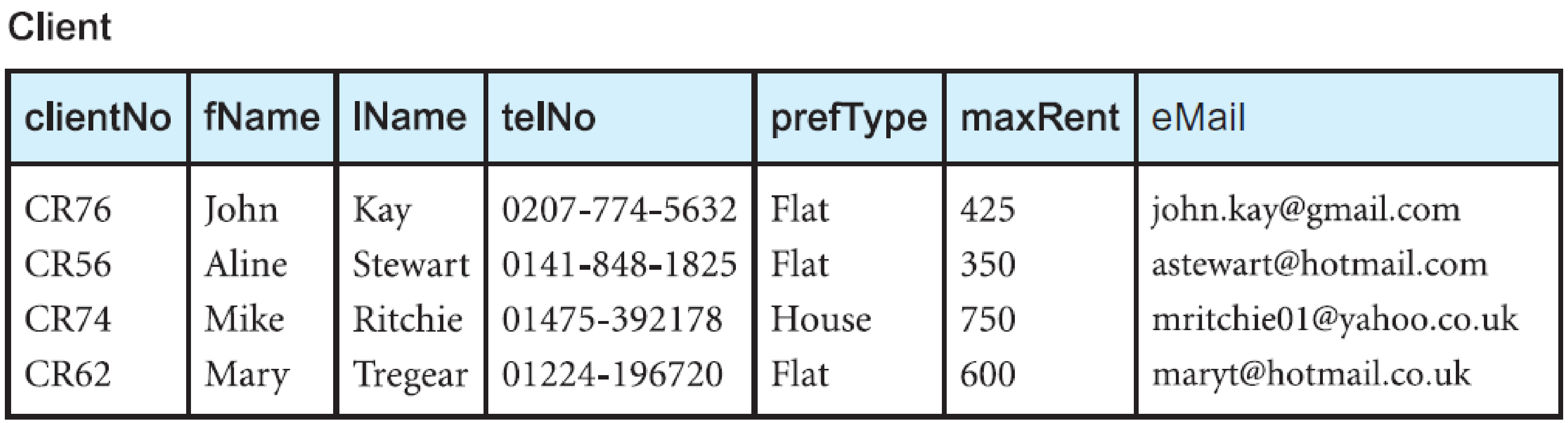
P_CODE	P_DESCRIPT	PRICE	STORE	AISLE	SHELF
123456	Flashlight	5.26	23	W	5
123456	Flashlight	5.26	24	K	9
123456	Flashlight	5.26	25	Z	6
123457	Lamp	25.15	23	W	5
123457	Lamp	25.15	24	K	9
123457	Lamp	25.15	25	Z	6
123458	Box Fan	10.99	23	W	5
123458	Box Fan	10.99	24	K	9
123458	Box Fan	10.99	25	Z	6
213345	9v battery	1.92	23	W	5
213345	9v battery	1.92	24	K	9
213345	9v battery	1.92	25	Z	6
311452	Powerdrill	34.99	23	W	5
311452	Powerdrill	34.99	24	K	9
311452	Powerdrill	34.99	25	Z	6
254467	100W bulb	1.47	23	W	5
254467	100W bulb	1.47	24	K	9
254467	100W bulb	1.47	25	Z	6



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	

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



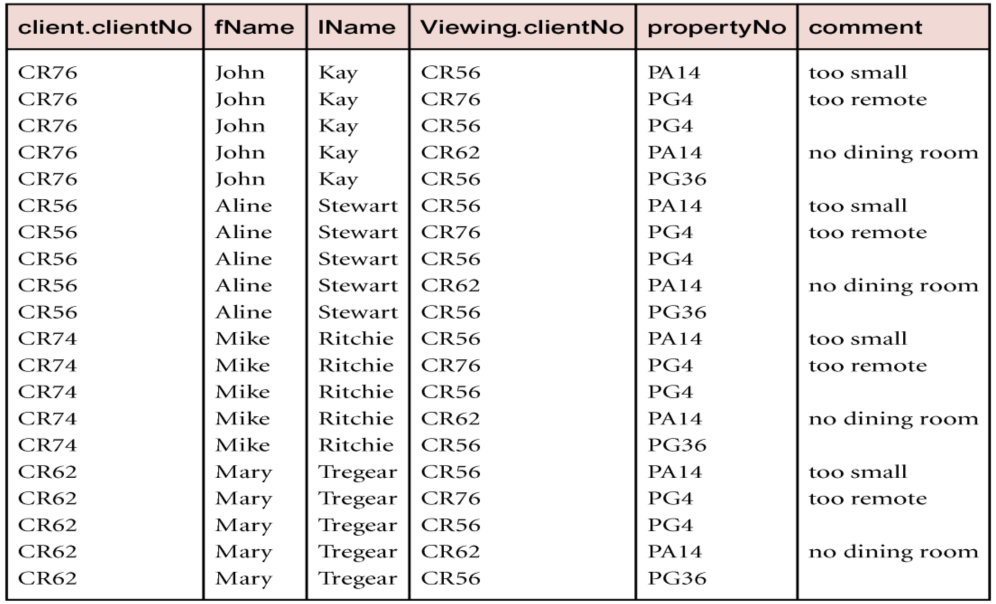
Client

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

Example - Cartesian Product

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

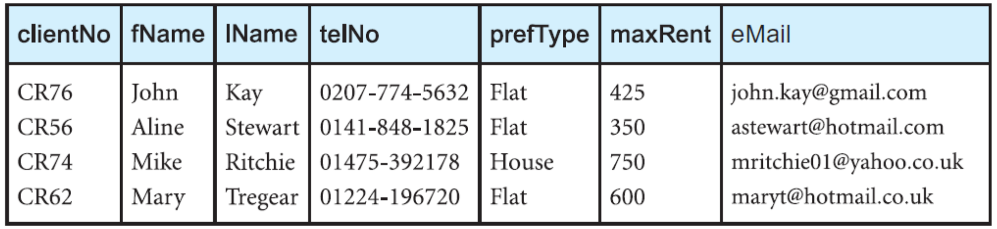
$$(\Pi_{\text{clientNo, fName, lName}} (\text{Client})) \text{ X } (\Pi_{\text{clientNo, propertyNo,comment}} (\text{Viewing}))$$



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	

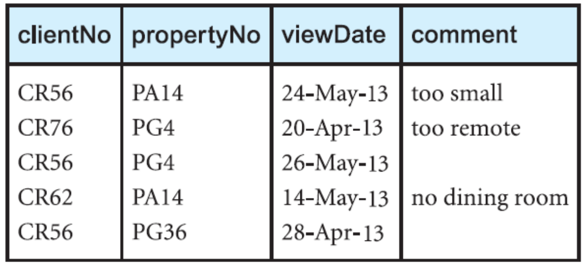
Client X Viewing

Client

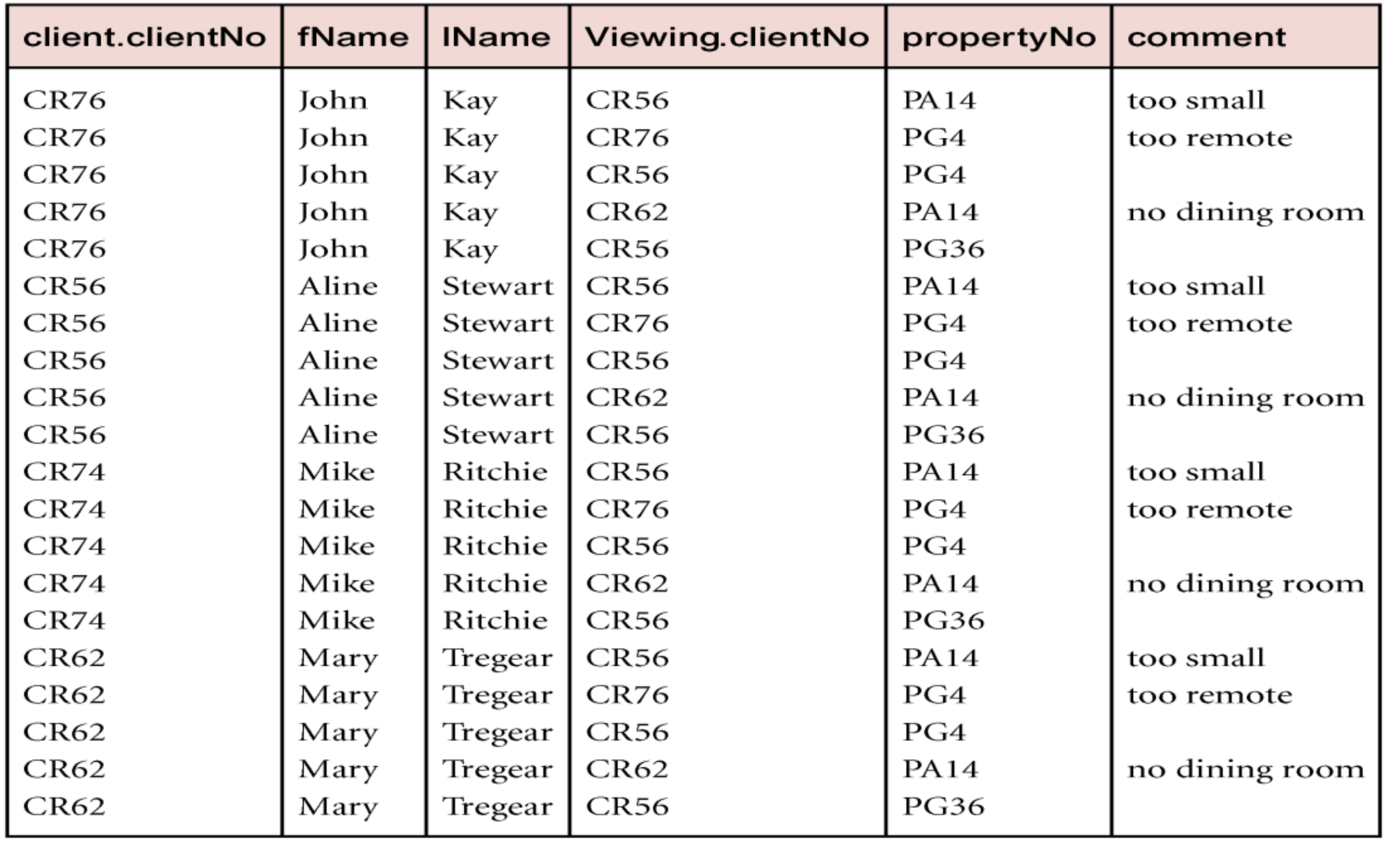


clientNo	fName	lName	telNo	prefType	maxRent	eMail
CR76	John	Kay	0207-774-5632	Flat	425	john.kay@gmail.com
CR56	Aline	Stewart	0141-848-1825	Flat	350	astewart@hotmail.com
CR74	Mike	Ritchie	01475-392178	House	750	mritchie01@yahoo.co.uk
CR62	Mary	Tregear	01224-196720	Flat	600	maryt@hotmail.co.uk

Viewing



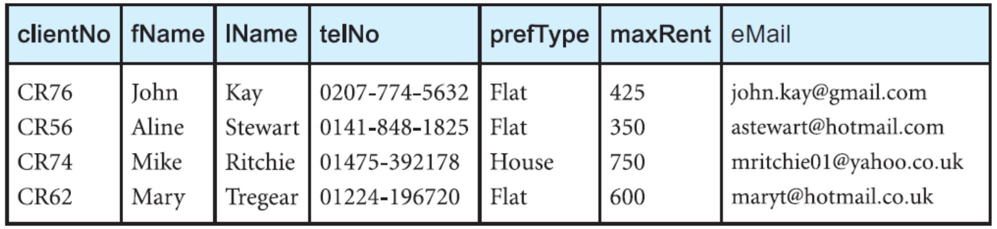
clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-13	too small
CR76	PG4	20-Apr-13	too remote
CR56	PG4	26-May-13	
CR62	PA14	14-May-13	no dining room
CR56	PG36	28-Apr-13	



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	

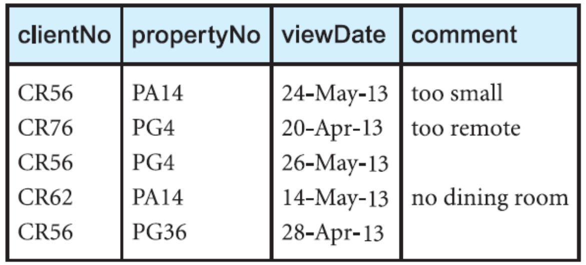
Client X Viewing with Select

Client

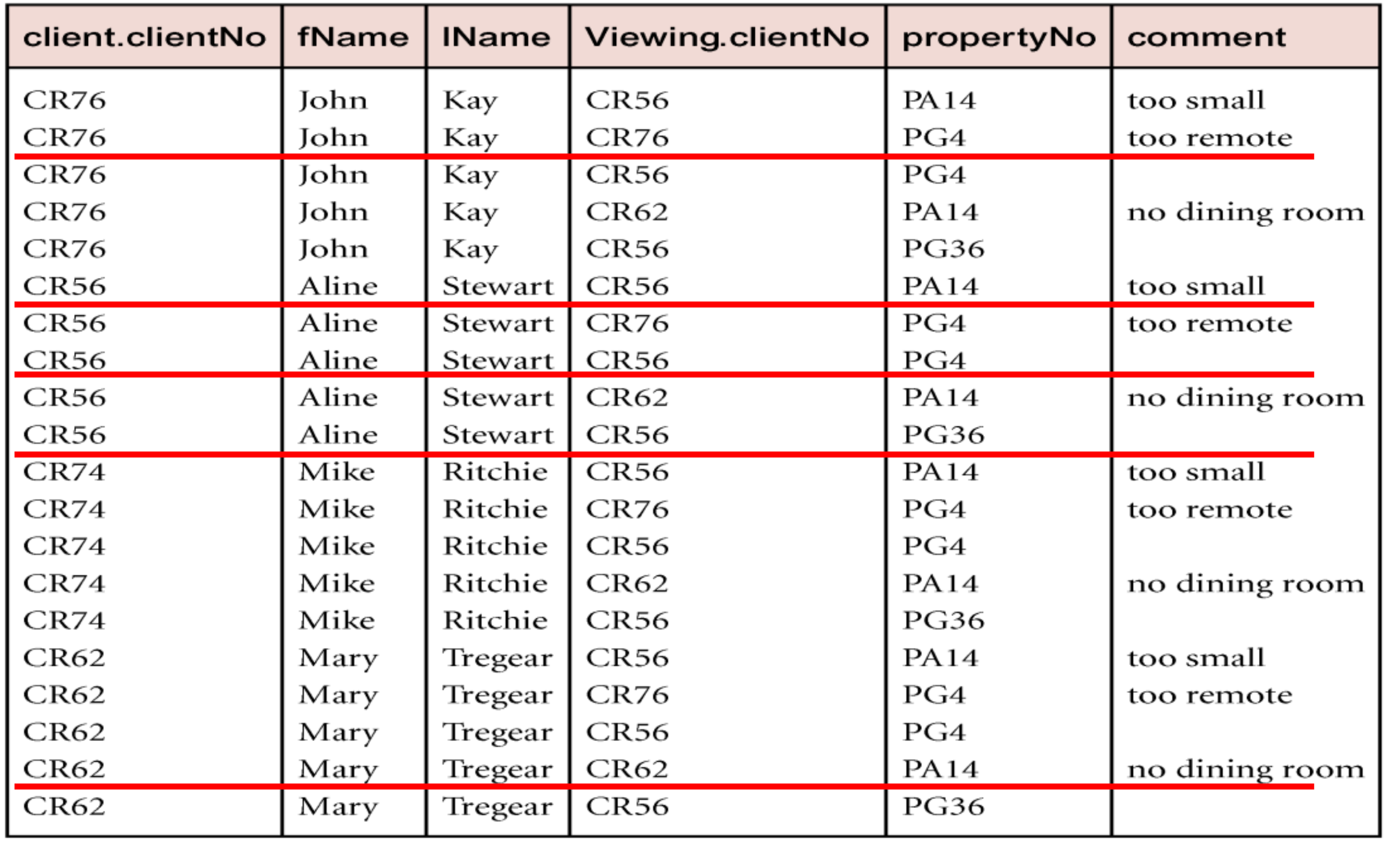


clientNo	fName	lName	telNo	prefType	maxRent	eMail
CR76	John	Kay	0207-774-5632	Flat	425	john.kay@gmail.com
CR56	Aline	Stewart	0141-848-1825	Flat	350	astewart@hotmail.com
CR74	Mike	Ritchie	01475-392178	House	750	mritchie01@yahoo.co.uk
CR62	Mary	Tregear	01224-196720	Flat	600	maryt@hotmail.co.uk

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	

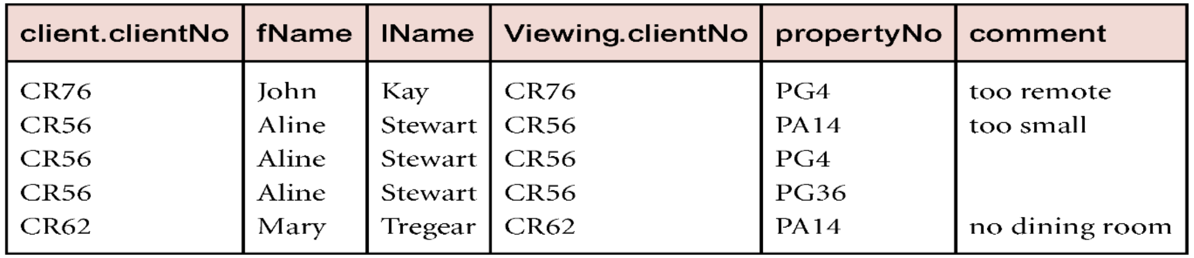


client.clientNo	fName	lName	Viewing.clientNo	propertyNo	comment
CR76	John	Kay	CR56	PA14	too small
CR76	John	Kay	CR76	PG4	too remote
CR76	John	Kay	CR56	PG4	
CR76	John	Kay	CR62	PA14	no dining room
CR76	John	Kay	CR56	PG36	
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR76	PG4	too remote
CR56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

Example - Cartesian Product and Selection

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

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



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

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

Division ÷ or /

- Derived operation - useful for expressing queries like: Find sailors who have reserved ***all or every*** boats.
- Let A have 2 fields, x and y; B have only field y:
  - A/B = $\left\{\langle x\rangle \mid \exists \langle x, y\rangle \in A \ \forall \langle y\rangle \in B\right\}$
  - i.e., ***A/B* contains all *x* tuples (sailors) such that for *every* *y* tuple (boat) in *B*, there is an *xy* tuple in *A*.**
- Or: If the set of y values (boats) associated with an x value (sailor) in A contains all y values in B, the x value is in A/B.

Division

- one single-column table (i.e. column “y”) as the divisor
- one 2-column table (i.e. columns “x” and “y”) as the dividend.
- *The tables must have a common column (i.e. column “y”.)*
- The output of the operation is a single column with the values of column “x” from the dividend table rows where the values of the common column (i.e. column “y”) in both tables match.

EXAMPLE

Retrieve all ERPs of students enrolled in every course.

ERP	CID
101	DB
102	OS
103	DB
103	OS

Table Name: Enrollment

CID	CName
DB	Database Systems
OS	Operating Systems

Table Name: Courses

Notation: A(X,Y)/B(X)

Enrollment(SID,CID)/Course(CID)

$$\Big((\Pi_{ERP,cid}(Enrollment)) \,/\, (\Pi_{cid}(Course))\Big)$$

Step 1: All students in enrolled in all courses

Notation: A(X,Y)/B(X)

Enrollment(ERP,CID)/Course(CID)

$$\left((\Pi_{ERP}(Enrollment))\ \mathrm{X}\ (\Pi_{cid}(Course))\right)$$

ERP
101
102
103

CID
DB
OS

ERP	CID
101	DB
101	OS
102	DB
102	OS
103	DB
103	OS

Step 2: Students not enrolled in every course

$$\left((\Pi_{ERP}(Enrollment)) X\ (\Pi_{cid}(Course)) \right) - Enrollment$$

ERP	CID
101	DB
101	OS
102	DB
102	OS
103	DB
103	OS

ERP	CID
101	DB
102	OS
103	DB
103	OS

Result:

ERP	CID
101	OS
102	DB

Step 3: Students enrolled in every course

$$(\Pi_{ERP}(Enrollment)) \text{ - } ((\Pi_{ERP}\big((\Pi_{ERP}(Enrollment))X\ (\Pi_{cid}(Course)) - Enrollment\ \big)))$$

ERP
101
102
103

ERP
101
102

Result:

ERP
103

Examples of Division A/B

sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

A

pno
p2

B1

pno
p2
p4

B2

pno
p1
p2
p4

B3

A/B1

A/B2

A/B3