Relational Algebra

Relational Operators

- Manipulating Data
  - The Relational Data Model is designed so that data may be processed with mathematical operations.
  - In maths an operation is a procedure that produces a new value from one or more input values.
  - For example: Projection

Relational Operators

PRODUCT CODE	PRODUCT NAME	UNIT PRICE	REMARKS
101	MELON	800G	WITH SEEDS
102	STRAWBERRY	150G	
103	APPLE	120G	
104	LEMON	200G	SOUR
201	CHESTNUT	100G	WITH BUR
202	PERSIMMON	160G	
301	PEACH	130G	
302	KIWI	200G	VALUABLE

PRODUCT NAME
MELON
STRAWBERRY
APPLE
LEMON
CHESTNUT
PERSIMMON
PEACH
KIWI

- We can extract the product name
- An operation to extract a column like this is called **projection**

Relational algebra

- Relational algebra and relational calculus are formal languages associated with the relational model.
- Informally, relational algebra is a (high-level) procedural language and relational calculus a non-procedural language.
- However, formally both are equivalent to one another.
- A language that produces a relation that can be derived using relational calculus is relationally complete.

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.

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 be expressed in terms of 5 basic operations

Relational Algebra

- Four relational operations
  - Projection
  - Selection
  - Join
  - Division
- Four set operations
  - Union
  - Difference
  - Intersection
  - Cartesian Product

Algebra

- Algebra
  - “The branch of mathematics concerning the study of the rules of operations and relations.”
- Operations and Operators
  - “An action or procedure which produces a new value from one or more input values.”
  - For example:+ - ! ÷
  - Take numbers as input and return numbers as results
  - Some operations have restrictions
  - For example you can’t divide by zero

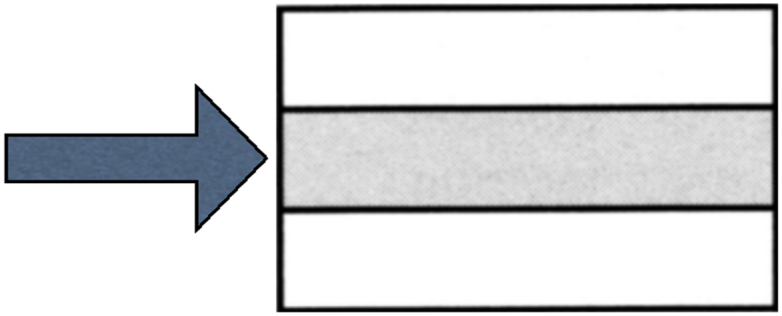
Relational Algebra

- Relational operators take one or two **relations** as inputs and return relations as the result.
- Set operators take one or two **sets** as inputs and return sets as the result.

Selection (or Restriction)

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

Product name	Unit price
Melon	800G
Strawberry	150G



Selection Example

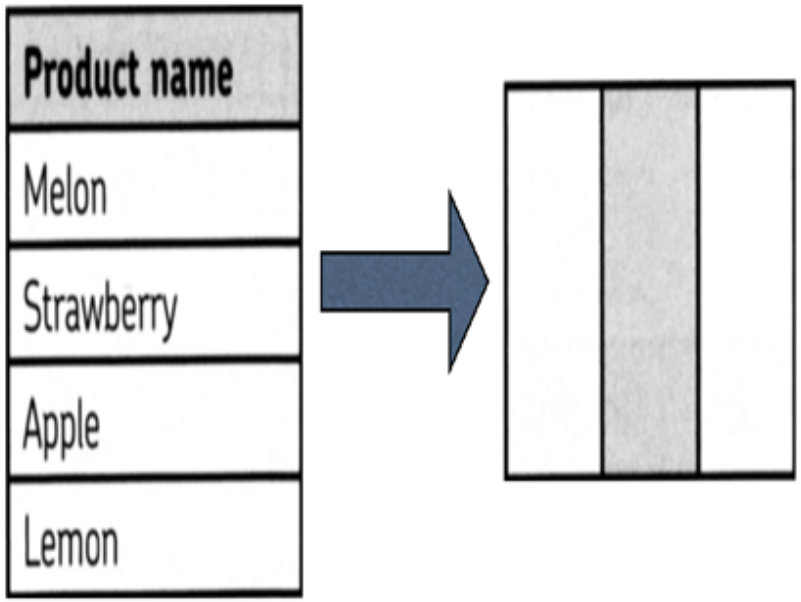
- *List all staff with a salary greater than £10,000.*
  - $\sigma_{salary > 10000}(Staff)$

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

- More complex predicates can be generated using the logical operators $\wedge$ (AND), $\vee$ (OR) and ~ (NOT).

Projection

- Works on a single relation and defines a relation that contains a vertical subset of R (i.e. as a column)
- Extracting the values of specified attributes and eliminating duplicates.
- $\Pi_{col1, \ldots, coln}(R)$



Projection Example

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

Union

- 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.
- $R \cup S$
- 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

- **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.
- Note that attributes names are not used in defining union- compatibility.
- In some cases, the Projection operation may be used to make two relations union-compatible.

Union Example 1

Product table 1 ∪ Product table 2

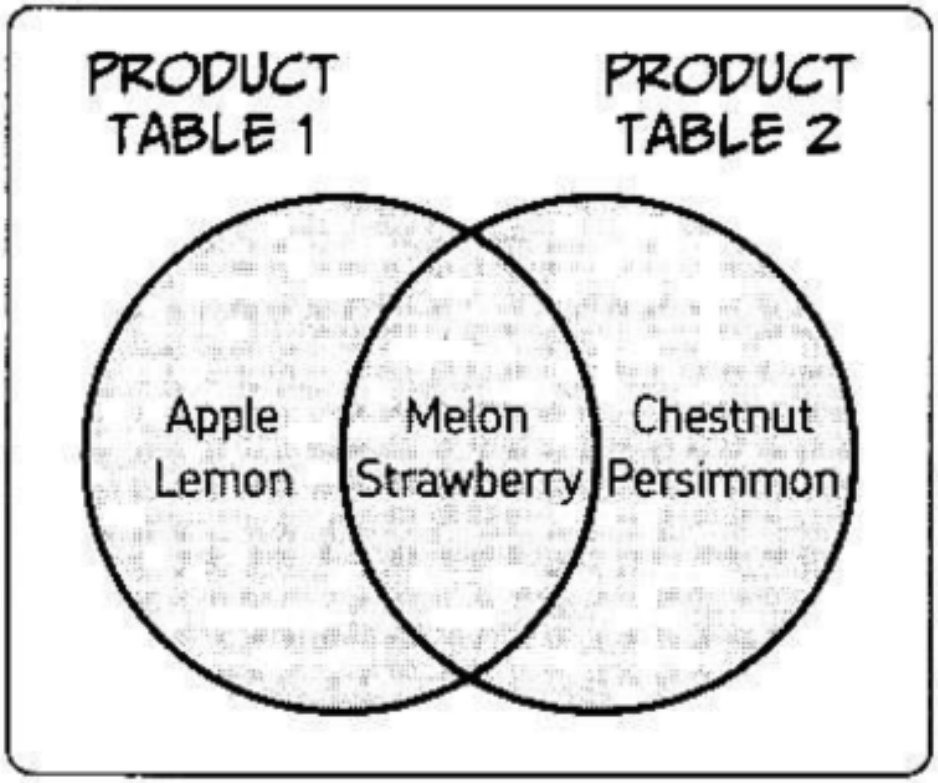
Product name	Unit price
Melon	800G
Strawberry	150G
Apple	120G
Lemon	200G

!

Product name	Unit price
Melon	800G
Strawberry	150G
Chestnut	100G
Persimmon	350G

Product name	Unit price
Melon	800G
Strawberry	150G
Apple	120G
Lemon	200G
Chestnut	100G
Persimmon	350G

Union Example 1



Union Example 2

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

Union Example 2

- Πcity(Branch) ∪ Πcity(PropertyForRent)

city
London
Aberdeen
Glasgow
Bristol

Set difference

- 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.
- R – S

Set difference Example 1

- *all of the products in the first table that don't appear in the second.*
  - Product table 1 – Product table 2

Product name	Unit price
Melon	800G
Strawberry	150G
Apple	120G
Lemon	200G

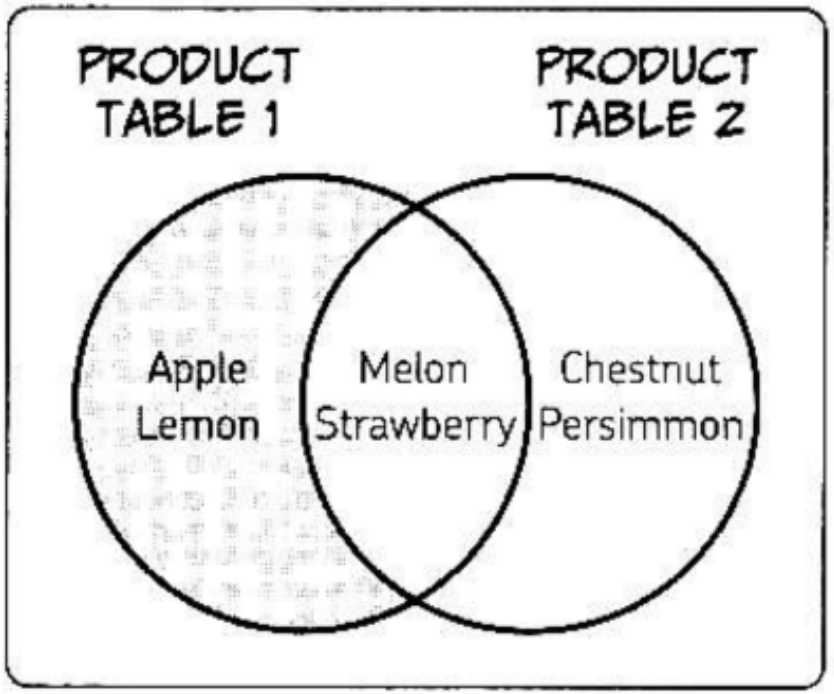
–

Product name	Unit price
Melon	800G
Strawberry	150G
Chestnut	100G
Persimmon	350G

Product name	Unit price
Apple	120G
Lemon	200G

Set difference Example 1

- Product table 1 – Product table 2



Set difference Example 2

- *For example - all of the products in the second table that don't appear in the first.*
- Product table 2 – Product table 1

Product name	Unit price
Melon	800G
Strawberry	150G
Apple	120G
Lemon	200G

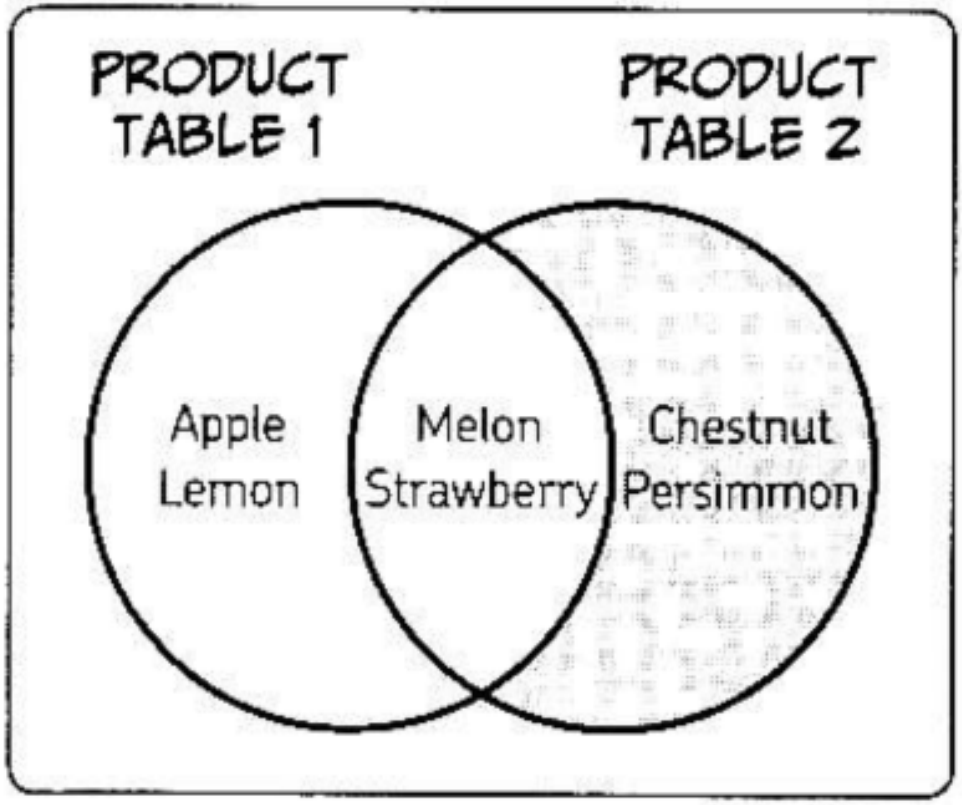
–

Product name	Unit price
Melon	800G
Strawberry	150G
Chestnut	100G
Persimmon	350G

Product name	Unit price
Chestnut	100G
Persimmon	350G

Set difference Example 2

- Product table 2 – Product table 1



Set difference Example 3

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

city
Bristol

Intersection

- The Intersection operation defines a relation consisting of the set of all tuples that are in both R and S.
- R and S must be union-compatible.
- $R \cap S$
- Expressed using basic operations:
  - $R \cap S = R - (R - S)$

Intersection Example 1

- *Extract items that are included in both tables*
  - Product table 1 ∩ Product table 2

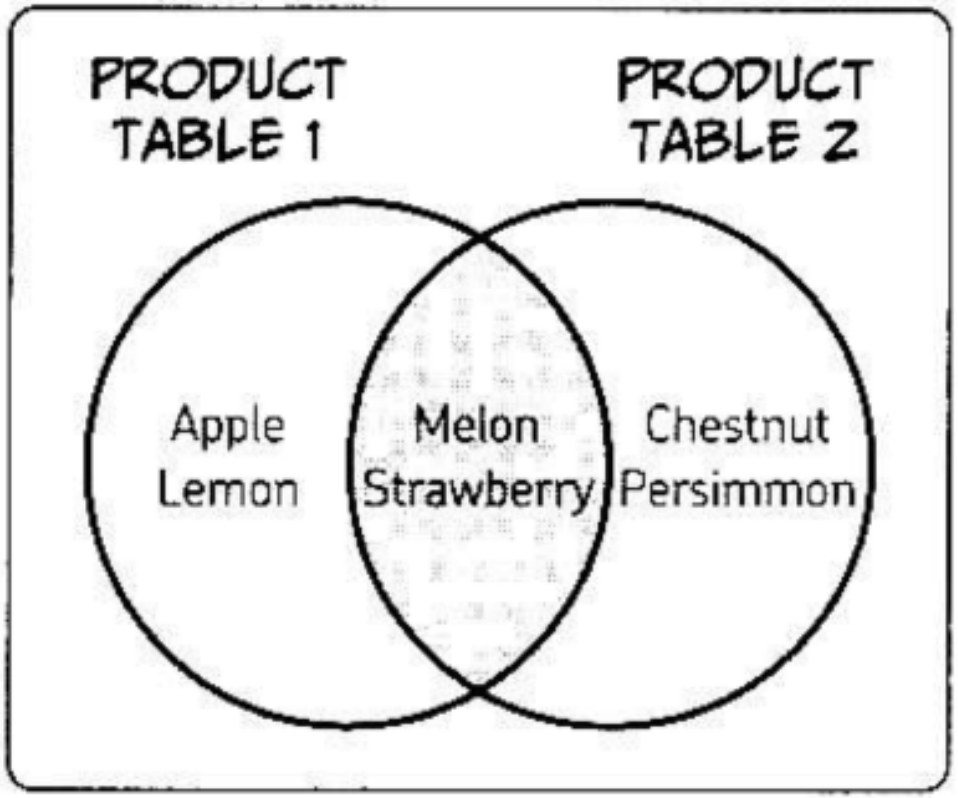
Product name	Unit price
Melon	800G
Strawberry	150G
Apple	120G
Lemon	200G

"

Product name	Unit price
Melon	800G
Strawberry	150G
Chestnut	100G
Persimmon	350G

Product name	Unit price
Melon	800G
Strawberry	150G

Intersection Example 1



Intersection Example 2

- *List all cities where there is both a branch office and at least one property for rent.*
  - Πcity(Branch) ∩ Πcity(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.
- It is possible that the two relations may have attributes with the same name.
  - In this case, the attribute names are prefixed with the relation name to maintain the uniqueness of attribute names within a relation.

Cartesian product Example 1

- *Combines all rows in the two tables.*

Product code	Product name	Unit price
101	Melon	800G
102	Strawberry	150G
103	Apple	120G

X

Export dest. code	Export dest. name
12	The Kingdom of Minanmi
23	Alpha Empire
25	The Kingdom of Ritol

3 rows

Product code	Product name	Unit price	Export dest. code	Export dest. name
101	Melon	800G	12	The Kingdom of Minanmi
101	Melon	800G	23	Alpha Empire
101	Melon	800G	25	The Kingdom of Ritol
102	Strawberry	150G	12	The Kingdom of Minanmi
102	Strawberry	150G	23	Alpha Empire
102	Strawberry	150G	25	The Kingdom of Ritol
103	Apple	120G	12	The Kingdom of Minanmi
103	Apple	120G	23	Alpha Empire
103	Apple	120G	25	The Kingdom of Ritol

3 x 3 =
9 rows

Cartesian product Example 2

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

Client

clientNo	fName	lName	telNo	prefType	maxRent
CR76	John	Kay	0207-774-5632	Flat	425
CR56	Aline	Stewart	0141-848-1825	Flat	350
CR74	Mike	Ritchie	01475-392178	House	750
CR62	Mary	Tregear	01224-196720	Flat	600

Viewing

clientNo	propertyNo	viewDate	comment
CR56	PA14	24-May-04	too small
CR76	PG4	20-Apr-04	too remote
CR56	PG4	26-May-04	
CR62	PA14	14-May-04	no dining room
CR56	PG36	28-Apr-04	

Cartesian product Example 2

- (ΠclientNo, fName, lName(Client)) × (ΠclientNo, 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 2

- Problems
  - this relation contains more information than we require.
  - For example, the first tuple of this relation contains different clientNo values.
  - To obtain the required list, we need to carry out a Selection operation on this relation to extract those tuples where

    Client.clientNo = Viewing.clientNo

Cartesian product and Selection Example 2

- σClient.clientNo = Viewing.clientNo((ΠclientNo, fName, lName(Client)) × (ΠclientNo, propertyNo, comment(Viewing)))

- Restricted Cartesian product of reduced Client and Viewing relations.

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*.

Join

- 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.

Join

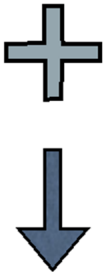
- Joins two tables together using a shared key (i.e. primary/foreign)

Product Table

Product code	Product name	Unit price
101	Melon	800G
102	Strawberry	150G
103	Apple	120G
104	Lemon	200G

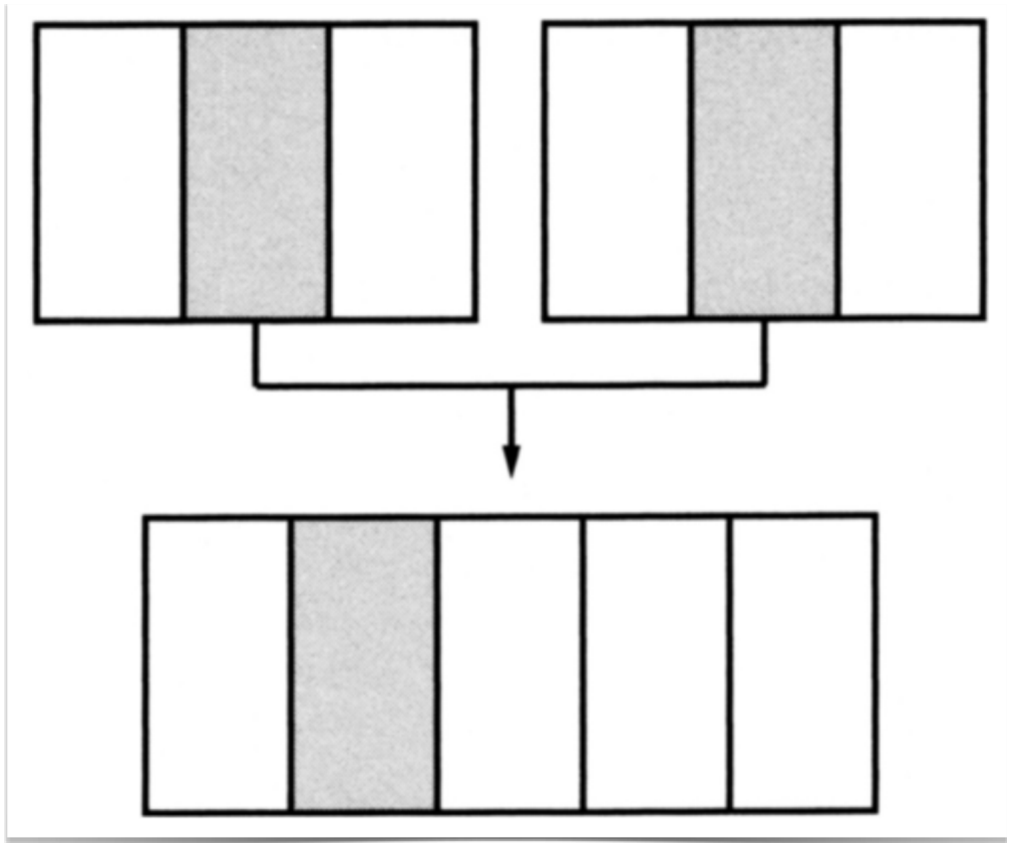
Sales Table

Date	Product code	Quantity
11/1	102	1,100
11/1	101	300
11/5	103	1,700
11/8	101	500



Date	Product code	Product name	Unit price	Quantity
11/1	102	Strawberry	150G	1,100
11/1	101	Melon	800G	300
11/5	103	Apple	120G	1,700
11/8	101	Melon	800G	500

Join



Join

- Various forms of join operation
  - Theta join
  - Equijoin (a particular type of Theta join)
  - Natural join
  - Outer join
  - Semijoin

Theta join (q-join)

- R $\bowtie$ FS
- 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 S.b_i where may be one of the comparison operators (<, , >, , =,).

Equijoin Example

- *List the names and comments of all clients who have viewed a property for rent*.
- (clientNo, fName, lName(Client)) ⋈ Client.clientNo = Viewing.clientNo (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 (Inner Join)

- R ⋈ 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 (Inner Join)

- Usually called JOIN
- JOIN is a binary operator that combines the PRODUCT and SELECT operators into a single operation.
- The result of the natural join of A and B is the combinations of tuples a from A and b from B where certain conditions are satisfied

Natural join (Inner Join) Example

- *List the names and comments of all clients who have viewed a property for rent.*
- clientNo, fName, lName(Client)) ⋈(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

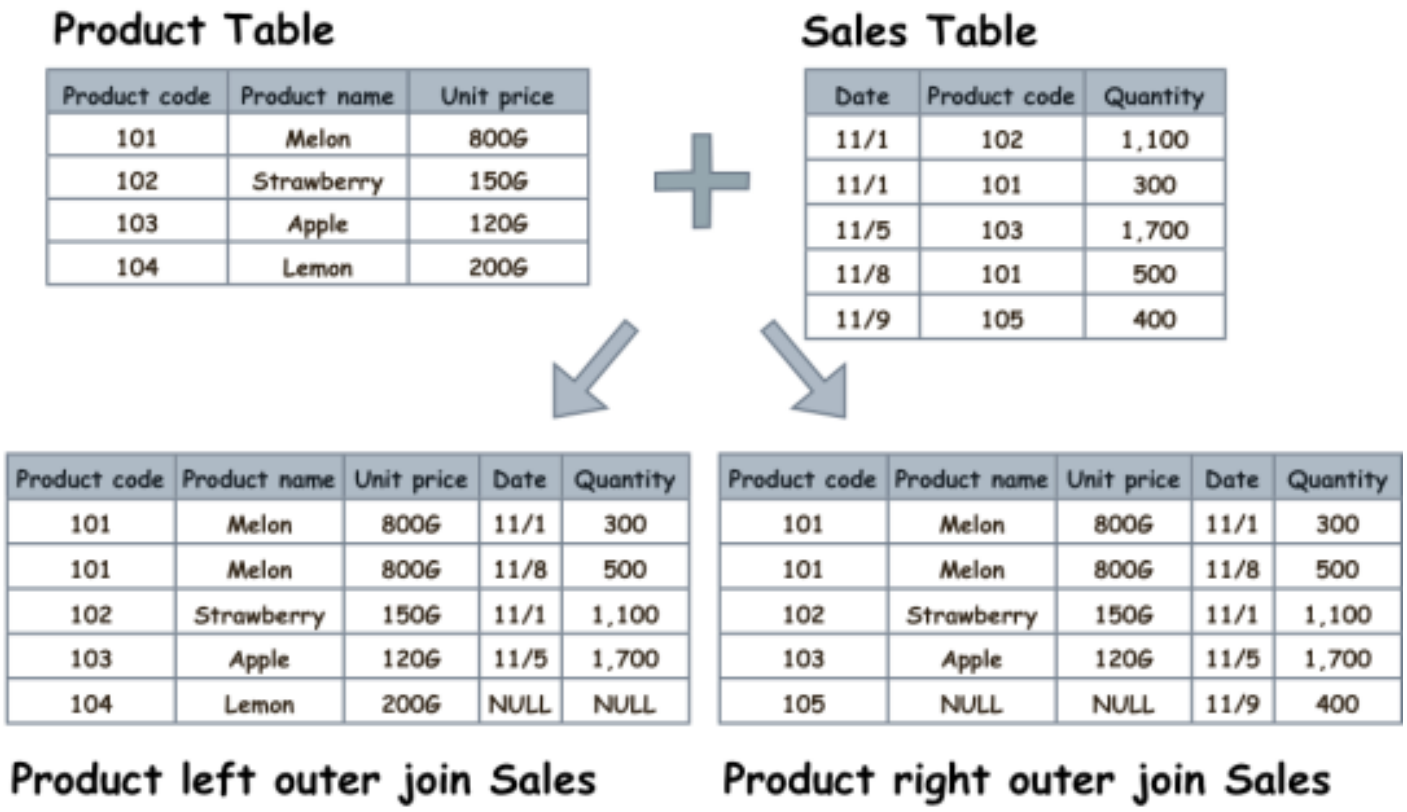
Outer join

- To display rows in the result that do not have matching values in the join column, use Outer join.
- R ⟕ S
- (Left) outer join is 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

- Outer join may retain the tuples which would be lost by natural join
  - Left outer join of A and B retains all of A
  - Right outer join of A and B retains all of B

Outer join Example



Left Outer join Example

- *Produce a status report on property viewings.*
- 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

- $R \triangleright_F S$
  - Defines a relation that contains the tuples of R that participate in the join of R with S.
- Can rewrite Semijoin using Projection and Join:
  - $R \triangleright_F S = A(R \bowtie_F S)$

Semijoin Example

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

  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

Division

- R S
- Defines a relation over the attributes C that consists of set of tuples from R that match combination of every tuple in S.
- Extracts rows whose column values match those in the second table, but only returns columns that don’t exist in the second table.

Division Example 1

Sales Table

Export dest. code	Export dest. name	Date
12	The Kingdom of Minanmi	3/5
12	The Kingdom of Minanmi	3/10
23	Alpha Empire	3/5
25	The Kingdom of Ritol	3/21
30	The Kingdom of Sazanna	3/25

÷

Export Destination Table

Export dest. code	Export dest. name
12	The Kingdom of Minanmi
23	Alpha Empire

The result of the division will be the date on which there are exports to both The Kingdom of Minanmi and the Alpha Empire.

Date
3/5

Division Example 2

- *Identify all clients who have viewed all properties with three rooms.*
- (clientNo, propertyNo(Viewing))
  (propertyNo(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