Electronics and
Computer Science

UNIVERSITY OF
Southampton

Relational Algebra

COMP3211 Advanced Databases

Nicholas Gibbins - nmg@ecs.soton.ac.uk
2016-2017

UNIVERSITY OF Southampton

The Relational Model

- Proposed by E.F. Codd in 1970
- Views data as mathematical relations
- Given sets S1, S2, …, Sn

  R is a relation on these n sets if it is a set of n-tuples each of which has its first element from S1, its second from S2, and so on

UNIVERSITY OF Southampton

Example Relation

Consider a database of student transcripts...

The relation schema TRANSCRIPT has three attributes:

- StudentID
- CourseCode
- Mark

The domains of these attributes are respectively:

- The set of integers
- The set of module codes {COMP1001, ..., COMP6031}
- The set of integers in the range 0..100

UNIVERSITY OF Southampton

Example Relation

A particular relation T of the relation schema TRANSCRIPT could be:

T = { <1, COMP2004, 77>,
<2, COMP2004, 76>,
<3, COMP1008, 44>,
<2, COMP1008, 66> }

Nomenclature

- A *relation* can be viewed as a table with columns and rows
- An *attribute* is a named column of a relation
- A *domain* is the set of allowable values for an attribute
- A *tuple* is a row of a relation (ordered collection of values)
- The *degree* of a relation is the number of attributes it contains
- The *cardinality* of a relation is the number of tuples it contains

UNIVERSITY OF Southampton

Properties of Relations

- Each relation in a relational database schema has a distinct name
- Each attribute of each tuple in a relation contains a single atomic value
- Each attribute has a distinct name
- The values of an attribute are all from the same domain
- Each tuple is distinct; there are no duplicate tuples
- The order of the attributes in a tuple is insignificant
- The order of the tuples in a relation is insignificant

University of Southampton

The Relational Algebra

A theoretical language which defines operations that allow us to construct new relations from other relations

Both the operands and the results of the operations are relations

Operations:

- Selection, projection, renaming
- Union, intersection, set difference
- Cartesian product
- Join (theta, natural, equi-, outer, semi-)
- Division

Why use the Relational Algebra?

Allows us to analyse the behaviour of complex queries in terms of fundamental operations

- More efficient queries
- More efficient database systems

A given SQL query may correspond to a number of different relational algebra expressions (query trees)

- Intermediate relations may have different sizes
- Cost of evaluation may vary
- Basis for query optimisation

UNIVERSITY OF Southampton

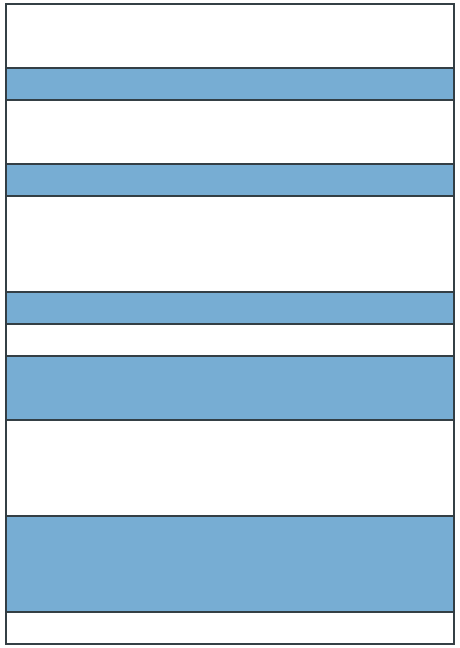
Relational Operators

Selection (or Restriction)

A unary operation on a relation R that defines a relation which contains only those tuples of R that satisfy the predicate

Selection extracts rows from a table

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



UNIVERSITY OF Southampton

Selection example

Find all the tuples in Staff for which salary is at least 35000

Staff

staffno	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

University of Southampton

Selection example

$$\sigma_{\text{salary} \geq 35000}(\text{Staff})$$

staffno	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

UNIVERSITY OF Southampton

Predicates

Atomic predicates take one of two forms:

- $a \theta b$
- $a \theta v$

a ,b are attribute names,

- θ is a binary operator from the set $\{\leq, <, =, >, \geq\}$
- v is a value constant

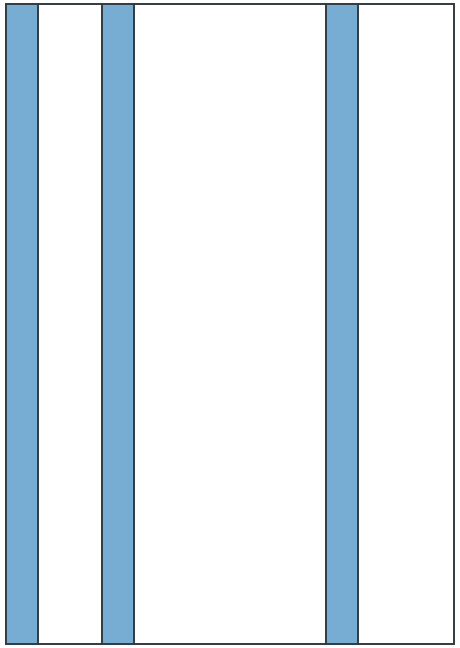
Predicates may be combined using the logical operators:

$\neg, \vee, \wedge$

UNIVERSITY OF Southampton

Projection

A unary operation on a relation R which defines a relation (a vertical subset of R) containing the specified attributes



Projection extracts columns from a table

$$\pi_{a_1,\ldots,a_n}(R)$$

$$\pi_A(R),\ \ A = \{a_1, \ldots, a_n\}$$

UNIVERSITY OF Southampton

Projection example

Find the staffno and salary for all tuples in Staff

Staff

staffno	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

University of Southampton

Projection example

$$\pi_{\text{staffno,salary}}(\text{Staff})$$

staffno	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

UNIVERSITY OF Southampton

Renaming

A unary operation on a relation R which renames the attributes of R

The rename specification is written as new name/old name

Renaming does not change the shape of a table

$$\rho_{a'/a}(R)$$

UNIVERSITY OF Southampton

Renaming example

Find all tuples of Staff with staffno renamed to id

Staff

staffno	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

UNIVERSITY OF Southampton

Renaming example

$$\rho_{\mathrm{id/staffno}}(\mathrm{Staff})$$

id	lname	fname	salary
102341	Smith	Adam	35000
104128	Jones	Edward	36850
106293	Smith	James	31250
114283	Brown	Fred	27000
128947	Archer	David	42000
134293	James	Edward	37500

UNIVERSITY OF Southampton

Decomposing Complex Operations

We can combine the relational algebra operations to build expressions of arbitrary complexity

For the sake of legibility, we can name intermediate expressions using the assignment operator

$$\pi_{\text{fname,lname}}(\sigma_{\text{salary}>40000}(\text{Staff}))$$

$$\begin{aligned} \text{ExpensiveStaff} &\leftarrow \sigma_{\text{salary}>40000}(\text{Staff}) \\ \text{Result} &\leftarrow \pi_{\text{fname,lname}}(\text{ExpensiveStaff}) \end{aligned}$$

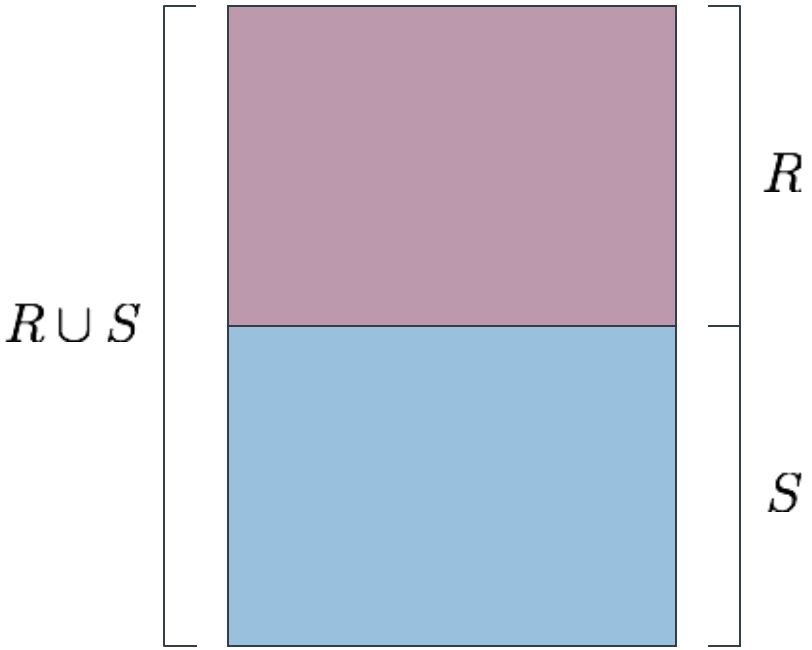
UNIVERSITY OF Southampton

Set Union

The union of two relations R and S contains all the tuples of R and S (eliminating duplicates).

R and S must have compatible schemas (same number of attributes, with the same domains)

$$R \cup S$$



Union Example

$$\sigma_{course=COMP1004 \wedge mark<40}(Transcript) \quad \cup$$

$$\sigma_{course=COMP1003 \wedge mark<40}(Transcript)$$

studentID	course	mark
40782385	COMP1004	67
40790324	COMP1004	38
40703913	COMP1004	37
40793128	COMP1004	59
40790324	COMP1003	34
40792837	COMP1003	64
40783002	COMP1003	32
40783041	COMP1003	73
40722823	COMP1003	61

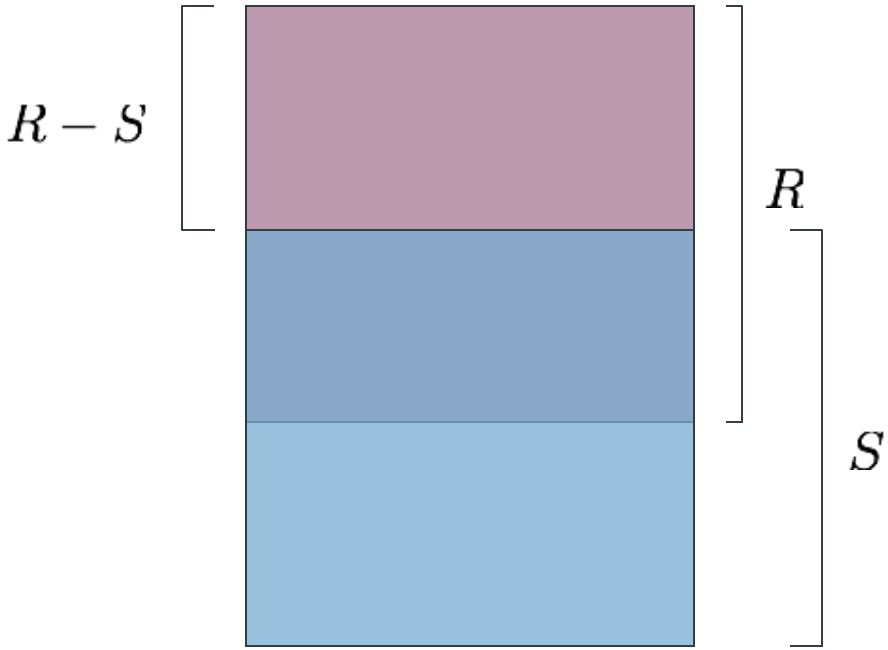
UNIVERSITY OF Southampton

Set Difference

The set difference of two relations R and S contains all of the tuples in R which are not in S

R and S must have compatible schemas (same number of attributes, with the same domains)

$$R - S$$



UNIVERSITY OF Southampton

Set Difference Example

$$\pi_{\text{studentID}}(\sigma_{\text{course}=\text{COMP1003}\wedge\text{mark}<40}(\text{Transcript})) \quad - \quad \pi_{\text{studentID}}(\sigma_{\text{course}=\text{COMP1004}\wedge\text{mark}<40}(\text{Transcript}))$$

studentID	course	mark
40782385	COMP1004	67
40790324	COMP1004	38
40703913	COMP1004	37
40793128	COMP1004	59
40790324	COMP1003	34
40792837	COMP1003	64
40783002	COMP1003	32
40783041	COMP1003	73
40722823	COMP1003	61

-

studentID	course	mark
40782385	COMP1004	67
40790324	COMP1004	38
40703913	COMP1004	37
40793128	COMP1004	59
40790324	COMP1003	34
40792837	COMP1003	64
40783002	COMP1003	32
40783041	COMP1003	73
40722823	COMP1003	61

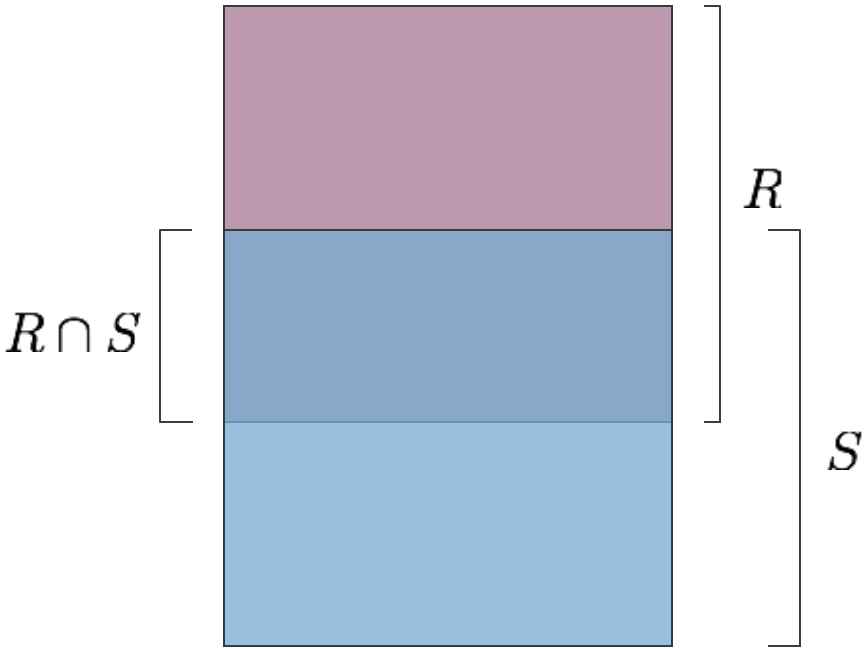
UNIVERSITY OF Southampton

Intersection

The intersection of two relations contains all the tuples that are in both relations

R and S must have compatible schemas (same number of attributes, with the same domains)

$$R \cap S$$



Intersection Example

$$\pi_{\text{studentID}}(\sigma_{\text{course}=\text{COMP1003}\wedge\text{mark}<40}(\text{Transcript})) \quad \cap$$

$$\pi_{\text{studentID}}(\sigma_{\text{course}=\text{COMP1004}\wedge\text{mark}<40}(\text{Transcript}))$$

studentID	course	mark
40782385	COMP1004	67
40790324	COMP1004	38
40703913	COMP1004	37
40793128	COMP1004	59
40790324	COMP1003	34
40792837	COMP1003	64
40783002	COMP1003	32
40783041	COMP1003	73
40722823	COMP1003	61

studentID	course	mark
40782385	COMP1004	67
40790324	COMP1004	38
40703913	COMP1004	37
40793128	COMP1004	59
40790324	COMP1003	34
40792837	COMP1003	64
40783002	COMP1003	32
40783041	COMP1003	73
40722823	COMP1003	61

UNIVERSITY OF Southampton

Cartesian Product

The cartesian product of two relations R and S contains the concatenation of every tuple in R with every tuple in S

R and S need not have compatible schemas

$$R \times S$$

R

a
b

S

1
2
3

$R \times S$

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

UNIVERSITY OF Southampton

Theta Join

The theta join of two relations contains those tuples from the cartesian product of the two relations which satisfy the predicate F

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \bowtie_{R.B=S.B} S$

A	B	C
a	1	x
a	1	y

$$R \bowtie_F S$$

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

Join Predicates

Predicates used in joins are of the form:

$$R.a \ \theta \ S.b$$

where a and b are attributes of R and S respectively and θ is one of $\{<, \leq, =, \geq, >, \neq\}$

UNIVERSITY OF Southampton

Equijoins and Natural Joins

An equijoin is a theta join in which the only predicate operator used is equality (=)

A natural join is an equijoin over all the common attributes of the joined relations; one occurrence of each common attribute is removed from the resulting relation

UNIVERSITY OF Southampton

Outer Join

The (left) outer join of two relations R and S is a join which also includes tuples from R which do not have corresponding tuples in S; missing values are set to null

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \rtimes_{R.B=S.B} S$

A	B	C
a	1	x
a	1	y
b	2	

$$R \rtimes S$$

$$R \rtimes S \equiv R \cup (R \bowtie S)$$

Outer Joins

The left outer join is not the only outer join

Right outer join $$R \ltimes S \equiv S \cup (R \bowtie S)$$

Full outer join $$R \rtimes\ltimes S \equiv R \cup S \cup (R \bowtie S)$$

UNIVERSITY OF Southampton

Full Outer Join Example

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \times_{R.B=S.B} S$

A	B	C
a	1	x
a	1	y
b	2	
	3	z

$$R \times S \equiv R \cup S \cup (R \bowtie S)$$

UNIVERSITY OF Southampton

Semijoin

A semijoin of two relations R and S contains all the tuples of R which are in the join of R and S with predicate F

R

A	B
a	1
b	2

S

B	C
1	x
1	y
3	z

$R \triangleright_{R.B=S.B} S$

A	B
a	1

$$R \triangleright_F S$$

$$R \triangleright_F S \equiv \pi_A(R \bowtie_F S)$$

University of Southampton

Mapping SQL to the Relational Algebra

```
SELECT *
FROM <table>
WHERE <pred>
```

$$\sigma_{pred}(table)$$

```
SELECT <cols>
FROM <table>
```

$$\pi_{cols}(table)$$

```
SELECT <cols>
FROM <table>
WHERE <pred>
```

$$\sigma_{pred}(\pi_{cols}(table))$$

$$\pi_{cols}(\sigma_{pred}(table))$$

```
SELECT <col> AS <col'>
FROM <table>
```

$$\rho_{col'/col}(\pi_{col}(table))$$

UNIVERSITY OF Southampton

Mapping Example

Consider a company personnel database

- Department relation
  - attributes dname, dnumber
  - dnumber is primary key
- Employee relation
  - attributes lname, fname, address, dno
  - dno is a foreign key for Department

UNIVERSITY OF Southampton

Mapping Example

Retrieve the name and address of all employees who work for the 'Research' department

```
SELECT FNAME, LNAME, ADDRESS
FROM       EMPLOYEE, DEPARTMENT
WHERE DNAME='Research' AND DNUMBER=DNO
```

$$\text{ResDept} \leftarrow \sigma_{dname=\text{'}Research\text{'}}(\text{Department})$$

$$\text{DeptEmps} \leftarrow \text{ResDept} \bowtie_{dumber=dno} \text{Employee}$$

$$\text{Result} \leftarrow \pi_{fname,lname,address}(\text{DeptEmps})$$

UNIVERSITY OF Southampton

Mapping Example

Retrieve the name and address of all employees who work for the ‘Research’ department

```
SELECT FNAME, LNAME, ADDRESS
FROM       EMPLOYEE, DEPARTMENT
WHERE DNAME=‘Research’ AND DNUMBER=DNO
```

$$\pi_{fname,lname,address}((\sigma_{dname='Research'}(\text{Department})) \bowtie_{dnumber=dno} \text{Employee})$$

UNIVERSITY OF Southampton

Mapping Example

Retrieve the name and address of all employees who work for the 'Research' department

```
SELECT FNAME, LNAME, ADDRESS
FROM       EMPLOYEE, DEPARTMENT
WHERE DNAME='Research' AND DNUMBER=DNO
```

$$\text{DeptEmps} \leftarrow \text{Department} \bowtie_{dumber=dno} \text{Employee}$$

$$\text{ResDept} \leftarrow \sigma_{dname=\text{'}Research\text{'}}(\text{DeptEmps})$$

$$\text{Result} \leftarrow \pi_{fname,lname,address}(\text{ResDept})$$

UNIVERSITY OF Southampton

Relational Transformations

Relational Transformations

Conjunctive selections cascade into individual selections

$$\sigma_{p \wedge q \wedge r}(R) = \sigma_p(\sigma_q(\sigma_r(R)))$$

Relational Transformations

Selection is commutative

$$\sigma_p(\sigma_q(R))) = \sigma_q(\sigma_p(R)))$$

Relational Transformations

Only the last in a sequence of projections is required

$$\Pi_L \Pi_M \ldots \Pi_N(R) = \Pi_L(R)$$

Relational Transformations

Selection and projection are commutative

(if the predicate only involves attributes in the projection list)

$$\Pi_{A_1,\ldots A_m}(\sigma_p(R)) = \sigma_p(\Pi_{A_1,\ldots A_m}(R))$$

Relational Transformations

Cartesian product and theta join are commutative

$$R \bowtie_p S = S \bowtie_p R$$

$$R \times S = S \times R$$

Relational Transformations

Selection distributes over theta join

(if the predicate only involves attributes being joined)

$$\sigma_p(R \bowtie_r S) = \sigma_p(S) \bowtie_r \sigma_p(R)$$

$$\sigma_p(R \times S) = \sigma_p(S) \times \sigma_p(R)$$

Relational Transformations

Projection distributes over theta join

(if projection list can be divided into attributes of the relations being joined, and join condition only uses attributes from the projection list)

$$\Pi_{L_1 \cup L_2}(R \bowtie_r S) = \Pi_{L_1}(R) \bowtie_r \Pi_{L_2}(S)$$

$$\Pi_{L_1 \cup L_2}(R \bowtie_r S) = \Pi_{L_1 \cup L_2}(\Pi_{L_1 \cup M_1}(R) \bowtie_r \Pi_{L_2 \cup M_2}(S))$$

Relational Transformations

Set union and intersection are commutative

$$R \cup S = S \cup R$$

$$R \cap S = S \cap R$$

Relational Transformations

Selection distributes over set operations

$$\sigma_p(R \cup S) = \sigma_p(R) \cup \sigma_p(S)$$

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

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

Relational Transformations

Projection distributes over set union

$$\Pi_L(R \cup S) = \Pi_L(R) \cup \Pi_L(S)$$

Relational Transformations

Associativity of theta join and cartesian product

$$(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$$

$$(R \times S) \times T = R \times (S \times T)$$

Relational Transformations

Associativity of set union and intersection

$$(R \cup S) \cup T = R \cup (S \cup T)$$

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

UNIVERSITY OF Southampton

Relational Algebra and Query Processing

UNIVERSITY OF Southampton

Why use the Relational Algebra, part 2?

Relation Algebra lets us:

- express queries in terms of underlying operations
- identify equivalent query plans
- transform query plans
- optimise query plans

Query Processing