

Basic Set Operations

- $r \cup s$
- $r \cap s$
- $r - s$

relation **r**

A	B	C
a	b	c
d	a	f
c	b	d

relation **s**

A	B	C
b	g	a
d	a	f

- Can we perform \cup , \cap , $-$ between any two relations?
 - They need to be *union compatible*
 - $|R| = |S|$ and
 - corresponding attributes have same domains
- Properties
 - Both \cup and \cap are commutative operations
 - Difference is not commutative

Attribute Names?

Basic Set Operations

- $r \cup s$
- $r \cap s$
- $r - s$

relation **r**

A	B	C
a	b	c
d	a	f
c	b	d

relation **s**

D	E	F
b	g	a
d	a	f

- Can we perform \cup , \cap , $-$ between any two relations?
 - They need to be *union compatible*
 - $|R| = |S|$ and
 - corresponding attributes have same domains
- Properties
 - Both \cup and \cap are commutative operations
 - Difference is not commutative

Attribute Names?

Cartesian Product

- $r \times s$

relation **r**

A	B	C
a	b	c
d	a	f
c	b	d

α_r

relation **s**

A	B
b	g
d	a

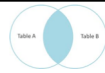
α_s

- Let $p(P) = r(R) \times s(S)$
- $|P| = ?$ and $|p| = ?$
 - $|P| = |R| + |S| = \alpha_r + \alpha_s$
 - $|p| = |r| * |s|$
- Name conflicts are resolved by using the relations names as prefixes: r.A, r.B, S.A, S.B

Common Query

- Library microDB:
 - Librarian (SSN, Name, SNO)
 - Section (SNO, SName, Head)
- List the names of head librarians.
- How?

Equi-Join



□ $r \bowtie_{r.A_i = s.A_j} S$

□ =-join is a macro of

$\sigma_{r.A_i = s.A_j}(r \times s)$

□ =-join of $r(R)$ and $s(S)$:

$r \bowtie_{r.B = s.D} S = ?$

relation s

C	D
3	4
6	8

relation r

A	B	C	D
1	2	3	4
2	4	6	8
1	2	4	8
2	6	6	8
8	2	3	4
2	4	3	4

Θ -Join

□ $\Theta = \{=, <, \leq, >, \geq, \neq\}$

□ Θ -join of $r(R)$ and $s(S)$
on attributes $r.A_i$ and $s.A_j$

$r \bowtie_{r.A_i \theta s.A_j} S$
 $= \sigma_{r.A_i \theta s.A_j}(r \times s)$

□ \geq -join of $r(R)$ and $s(S)$:

$r \bowtie_{r.B \geq s.D} S = ?$

relation r

A	B	C	D
1	2	3	4
2	4	6	8
1	2	4	8
2	6	6	8
8	2	3	4
2	4	3	4

relation s

C	D
3	4
6	8

Example of Θ -Join

□ \geq -join of $r(R)$ and $s(S)$:

$r \bowtie_{r.B \geq s.D} S = ?$

r.A	r.B	r.C	r.D	s.C	s.D
2	4	6	8	3	4
2	4	3	4	3	4
2	6	6	8	3	4

□ $r \bowtie s = r \times s, \Theta = \emptyset$

relation r

A	B	C	D
1	2	3	4
2	4	6	8
1	2	4	8
2	6	6	8
8	2	3	4
2	4	3	4

relation s

C	D
3	4
6	8

Natural-Join

□ Equi-join without duplicate columns

$r *_{\rho} S$

□ P =list of attributes: $P = R \cap S$

□ $r * s = \pi_{R \cup S}(r \bowtie_{r.P = s.P} s)$

□ $r * s = ?$

□ Note other notations & meanings

■ $r \bowtie s = r * s, R \cap S \neq \emptyset$

■ $r * s = r \times s, R \cap S = \emptyset$

relation r

A	B	C	D
1	2	3	4
2	4	6	8
1	2	4	8
2	6	6	8
8	2	3	4
2	4	3	4

relation s

C	D	E
3	4	6
6	8	8

Examples from Library DB

- Library DB schema:
 - LIBRARIAN(Name, SSN, SNO, BirthPlace)
 - SECTION(SName, SNO, Head)
 - OUTREACH(Pname, PNO, SNUM, Location)
 - WORKSON(LSSN, PNO, Hours)
- For every outreach activity located in PGH, list its project number, the responsible section and its head.
- Find all the librarians (Name, SSN) who work on projects located in their place of birth.

Division



- Let $r(R)$ and $s(S)$ be relations such as $S \subset R$

- The division of r by s , denoted by $r \div s$,
 - is relation whose schema is $Q = R - S$ and
 - includes all t such as $t_r[Q] = t$ and $t_r[S] = t$

relation r				relation s	
A	B	C	D	C	D
1	2	3	4	3	4
2	4	6	8	6	8
1	2	6	8		
2	4	3	4		

r ÷ s	
A	B
1	2
2	4

Division With Remainder



- Let $r(R)$ and $s(S)$ be relations such as $S \subset R$

- $r \div s$
 - is relation whose schema is $Q = R - S$
 - includes all t such as $t_r[Q] = t$ and $t_r[S] = t$

relation r				relation s	
A	B	C	D	C	D
1	2	3	4	3	4
2	3	3	5	6	8
2	4	6	8		
1	2	6	8		
8	2	3	4		
2	4	3	4		

r ÷ s	
A	B
1	2
2	4

Division Usage

- Query: “Retrieve the names of students who took *all* the classes that John took.”

Q = ENROLL ÷ CLASS

FN	LN
S	A
K	L

ENROLL

FN	LN	DP	No
S	A	CS	4
M	K	CS	5
K	L	EE	8
S	A	EE	8
F	S	CS	4
K	L	CS	4

CLASS

DP	No
CS	4
EE	8

Note:

Division can be expressed using π , \times and $-$ operations: $r \div s = \pi_Q(r) - \pi_Q((\pi_Q(r) \times s) - r)$

Division Usage: Review Example

- ◆ Query: “List the names of people who have in common *all* Susan’s friends on Facebook.”

Relation **Friends**

Username	FNFriend	LNFrind
Susan	Alex	L
Mark	Alex	L
Kirk	Mary	K
Shi	Alex	L
Susan	Mary	K
Shi	Lory	M
Kirk	Chia	S
Kirk	Alex	L

Division Usage: Review Example

- ◆ Query: “List the names of people who have in common *all* Susan’s friends on Facebook.”

Relation **Friends**

Username	FNFriend	LNFrind
Susan	Alex	L
Mark	Alex	L
Kirk	Mary	K
Shi	Alex	L
Susan	Mary	K
Shi	Lory	M
Kirk	Chia	S
Kirk	Alex	L

$SF \leftarrow \sigma_{\text{Username} = \text{'Susan'}}(\text{Friends})$

$SSF \leftarrow \text{Friends} \div \pi_{\text{FN, LN}}(SF)$

$RSLT \leftarrow \sigma_{\text{Username} \neq \text{'Susan'}}(SSF)$

Extended Relational Operations

- Extended set and Relational operations:
 - Outer Union
 - Outer Joins
- Aggregate operations:
 - MAX, MIN, AVG, SUM
 - Count
 - Subset: grouping
- Arithmetic operations and other functions:

Outer Union

- it is defined on partially union compatible relations
 - Non union-compatible attributes are kept in $r \cup^* s$
 - Non union-compatible attributes without value are set to NULL
 - Tuples are “matched” over common named attributes like in natural join

			$r \cup^* s$	
			FN	LN
relation r	relation s			
FN	LN	MJ	CL	
a	b	cs	Null	
d	a	ce	sr	
c	b	cs	Null	
c	b	cs		

- ◆ what about outer intersection or outer difference?

Outer Join

- Join selects only tuples satisfying the join-condition
- **Outer Join:**
 - Left outer join ($r \bowtie^L s$) keeps every tuple in the left relation
 - Right outer join ($r \bowtie^R s$) keeps every tuple in the right relation
 - Full outer join ($r \bowtie^F s$) keeps every tuple
- Attributes of tuples with no matching tuples are set to NULL
- With out a join-condition they behave like natural join

Outer Join ("natural")

relation r				relation s		
FN	LN	MJ	CL	FN	LN	CL
a	b	cs	Null	b	g	f
d	a	ce	sr	d	a	sr
c	b	cs	Null			

$r \bowtie^F s$			
FN	LN	MJ	CL
a	b	cs	Null
d	a	ce	sr
c	b	cs	Null
b	g	Null	f

Aggregate Functions

- Mathematical and Statistical aggregate functions on collections of values
 - SUM, MAXIMUM, MINIMUM, AVERAGE
 - COUNT number of tuples (cardinality)
- $f_{\langle \text{function list} \rangle}(\langle \text{relation} \rangle)$
 - *Function list* is a list of pairs (*< function, attribute >*)
- E.g., $f_{\text{count SID, AVERAGE GPA}}(\text{STUDENT})$

Aggregate Functions: Example

- $\text{RSLT} \leftarrow f_{\text{count SID, AVERAGE GPA}}(\text{STUDENT})$

Student

SID	Name	Age	GPA
546007	Susan	18	3.8
546100	Bob	19	3.65
546500	Bill	20	3.7

RSLT

Count_SID	AVERAGE_GPA
3	3.72

Example of Aggregation Query

❑ Q: Find the students with the highest GPA.

❑ **Student** (SID, Name, Age, GPA)

❑ A:

$MG(MGPA) \leftarrow f_{MAX\ GPA} (Student);$

$RSLT \leftarrow MG \bowtie_{MGPA = GPA} (Student);$

Grouping

❑ Grouping the tuple in a relation

$\langle \text{grouping attributes} \rangle f_{\langle \text{function list} \rangle} (\langle \text{relation} \rangle)$

▪ Tuples are grouped based on the values of *grouping attributes*

• E.g., $\text{major } f_{\text{count SID, AVERAGE GPA}} (STUDENT)$

major	Count_SID	AVERAGE_GPA
-------	-----------	-------------

Recursive closure

❑ It is applied to a recursive relationship between tuples of the same relation

❑ E.g., find all the ancestors or descendants

❑ How do we express it?

❑ What about the join operation?

❑ Need control statements...iteration

Write Queries in Relational Algebra

❑ Deletion:

▪ $r \leftarrow r - \text{Relational_Expression}$

▪ $STUDENT \leftarrow STUDENT - (\sigma_{\text{Dept} = 'CSD' \wedge \text{GPA} < 2.5} (STUDENT))$

❑ Insertion:

▪ $r \leftarrow r \cup \text{Relational_Expression}$

▪ $STUDENT \leftarrow STUDENT \cup \{(365, 'Smith', 'John')\}$

❑ Updating:

▪ $r \leftarrow \Pi_{\text{attributes-to-be-updated}} (r)$

▪ $STUDENT \leftarrow \Pi_{\text{Dept} = 'CSD'} (\sigma_{\text{Dept} = 'CS'} (STUDENT))$

Discussion

- ❑ The relational algebra is **procedural**
- ❑ The queries in relational algebra specify **how** to produce a result, BUT...
- ❑ The **how** should be the responsibility of the system
- ❑ User queries should be **declarative** specifying what is to be retrieved
 - Textual query languages (SQL, QUEL)
 - Graphical query languages (QBE)
 - Visual iconic languages (QBI)
- ❑ Other formal query languages:
 - Relational tuple calculus
 - Relational domain calculus

Steps in Processing a Query

