



Relational Algebra







Relational Algebra

- Relation Algebra is a Procedural Language
- Six basic operators
 - SELECT
 - PROJECT
 - UNION
 - SET DIFFERENCE
 - CARTISIAN PRODUCT
 - RENAME
- The operators take one or more relations as inputs and give a new relation as a result.







Select Operation – Example

• Relation *r*

Α	В	С	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

• $\sigma_{A=B \land D > 5}(r)$

Α	В	С	D
α	α	1	7
β	β	23	10







- □ Notation: $\sigma_p(r)$
- □ *p* is called the selection predicate
- Defined as:

$$\sigma_p(\mathbf{r}) = \{t \mid t \in r \text{ and } p(t)\}$$

Where p is a formula in propositional calculus consisting of terms connected by : \land (and), \lor (or), \neg (not) Each term is one of:

<attribute> op <attribute> or <constant>

where *op* is one of: =, \neq , \geq . \leq .

Example of selection:

$$\sigma_{\textit{branch-name}="Perryridge"}(\textit{account})$$







Project Operation



$$\prod_{A_1, A_2, \ldots, A_k} (r)$$

where A_1 , A_2 are attribute names and r is a relation name.

- The result is defined as the relation of k columns obtained by erasing the columns that are not listed
- Duplicate rows removed from result, since relations are sets
- E.g. To eliminate the *branch-name* attribute of *account* $\Pi_{account-number, \ balance}$ (account)







□ Relation *r*:

Α	В	C
α	10	1
α	20	1
β	30	1
β	40	2

 \square $\prod_{A,C} (r)$

		_		
Α	С		Α	С
α	1		α	1
α	1	=	β	1
β	1		β	2
β	2			





Union Operation

- □ Notation: $r \cup s$
- Defined as:

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

- For r ∪ s to be valid.
 - 1. *r*, *s* must have the *same arity* (same number of attributes)
 - 2. The attribute domains must be *compatible* (e.g., 2nd column of *r* deals with the same type of values as does the 2nd column of *s*)
- □ E.g. to find all customers with either an account or a loan $\Pi_{customer-name}$ (depositor) $\cup \Pi_{customer-name}$ (borrower)







Union Operation – Example

Relations *r*, *s*:

A	В
α	1
α	2
β	1
	r

Α	В	
α	2	
β	3	
S		

 $r \cup s$:

Α	В
α	1
α	2
β	1
β	3







Set Difference Operation

- Notation r s
- Defined as:

$$r-s = \{t \mid t \in r \text{ and } t \notin s\}$$

- Set differences must be taken between compatible relations.
 - ☐ *r* and *s* must have the *same arity*
 - □ attribute domains of *r* and *s* must be compatible





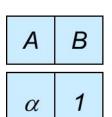


Set Difference Operation – Example





Relations *r*, *s*:



 α

А	D
α	2
β	3

r-	S
----	---





Cartesian-Product Operation-Example

Relations r, s:

Α	В	
α	1	
β	2	
r		

С	D	E
$\begin{array}{c} \alpha \\ \beta \\ \beta \\ \gamma \end{array}$	10 10 20 10	a a b b

r x s:

Α	В	С	D	E
α	1	α	10	а
α	1	β	10	а
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	а
β	2	β	10	а
β β β	2 2 2	$eta \ eta \ \gamma$	20	b
B	2	γ	10	b







Rename Operation





- Allows us to name, and therefore to refer to, the results of relational-algebra expressions.
- Allows us to refer to a relation by more than one name.
 Example:

$$\rho_X(E)$$

returns the expression *E* under the name *X*If a relational-algebra expression *E* has arity *n*, then

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

returns the result of expression E under the name X, and with the attributes renamed to A1, A2, ..., An.





Banking Example

branch (branch-name, branch-city, assets)

customer (customer-name, customer-street, customer-only)

account (account-number, branch-name, balance)

loan (loan-number, branch-name, amount)

depositor (customer-name, account-number)

borrower (customer-name, loan-number)









$$\sigma_{amount > 1200}$$
 (loan)

□ Find the loan number for each loan of an amount greater than \$1200

$$\prod_{loan-number} (\sigma_{amount > 1200} (loan))$$











Find the names of all customers who have a loan, an account, or both, from the bank

$$\Pi_{customer-name}$$
 (borrower) $\cup \Pi_{customer-name}$ (depositor)

Find the names of all customers who have a loan and an account at bank.

$$\Pi_{customer-name}$$
 (borrower) $\cap \Pi_{customer-name}$ (depositor)









Find the names of all customers who have a loan at the Perryridge branch.

$$\Pi_{customer-name}$$
 ($\sigma_{branch-name="Perryridge"}$ ($\sigma_{borrower.loan-number=loan.loan-number}$ (borrower x loan)))

Find the names of all customers who have a loan at the Perryridge branch but do not have an account at any branch of the bank.

$$\Pi_{customer-name}$$
 ($\sigma_{borrower.loan-number}$ = "Perryridge" ($\sigma_{borrower.loan-number}$ = loan.loan-number (borrower x loan))) - $\Pi_{customer-name}$ (depositor)







dge

Find the names of all customers who have a loan at the Perryridge branch.

Query 1

$$\Pi_{\text{customer-name}}(\sigma_{\text{branch-name}} = \text{``Perryridge''} (\sigma_{\text{borrower.loan-number}}(\sigma_{\text{borrower}} \times \sigma_{\text{borrower}}))$$

Query 2

$$\Pi_{customer-name}(\sigma_{loan.loan-number} = borrower.loan-number(\sigma_{branch-name} = "Perryridge"(loan)) x borrower))$$





Find the largest account balance

- □ Rename *account* relation as *d*
- The query is:

$$\Pi_{balance}(account)$$
 - $\Pi_{account.balance}$ $(\sigma_{account.balance} < d.balance (account x ρ_d (account)))$







Additional Operations





Y



We define additional operations that do not add any power to the relational algebra, but that simplify common queries.

- Set intersection
- Natural join
- Division
- Assignment





Set-Intersection Operation

- □ Notation: $r \cap s$
- Defined as:
- $\Box r \cap s = \{t \mid t \in r \text{ and } t \in s\}$
- Assume:
 - \square r, s have the same arity
 - attributes of r and s are compatible
- □ Note: $r \cap s = r (r s)$







Set-Intersection Operation - Example

Relation r, s:

Α	В
α	1
α	2
β	1

Α	В
α β	2 3

Α	В
α	2



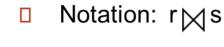














- Let r and s be relations on schemas R and S respectively. Then, $r \bowtie s$ is a relation on schema $R \cup S$ obtained as follows:
 - \square Consider each pair of tuples t_r from r and t_s from s.
 - □ If t_r and t_s have the same value on each of the attributes in $R \cap S$, add a tuple t to the result, where
 - \Box t has the same value as t_r on r
 - $\ \square \ t$ has the same value as t_S on s
- Example:

$$R = (A, B, C, D)$$

$$S = (E, B, D)$$

- ☐ Result schema = (A, B, C, D, E)
- \square $r \bowtie s$ is defined as:

$$\Pi_{r.A, r.B, r.C, r.D, s.E} (\sigma_{r.B = s.B} \wedge_{r.D = s.D} (r \times s))$$



Natural Join Operation – Example

B



Relations r, s:

Α	В	С	D
α	1	α	а
β	2	γ	а
γ	4	β	b
α	1	γ	а
δ	2	β	b
	r		

В	D	E
1 3 1 2 3	a a a b b	$\begin{array}{c} \alpha \\ \beta \\ \gamma \\ \delta \\ \in \end{array}$
	S	

 $r \bowtie s$

Α	В	С	D	Ε
α	1	α	а	α
α	1	α	а	γ
α	1	γ	а	α
α	1	γ	а	γ
δ	2	β	b	δ













$$r \div s$$

- Suited to queries that include the phrase "for all".
- Let r and s be relations on schemas R and S respectively where

$$\square$$
 $R = (A_1, ..., A_m, B_1, ..., B_n)$

$$\Box$$
 $S = (B_1, ..., B_n)$

The result of $r \div s$ is a relation on schema

$$R - S = (A_1, ..., A_m)$$

$$r \div s = \{ t \mid t \in \prod_{R-S}(r) \land \forall u \in s (tu \in r) \}$$





Division Operation - Example

Relations *r*, *s*:

В	
1	
2	
3	
1	
1	
1	
3	
4	
	1 2 3 1 1 1 3

1 2 s





Assignment Operation

B



-)
- 1

- □ The assignment operation (←) provides a convenient way to express complex queries.
 - Write query as a sequential program consisting of
 - □ a series of assignments
 - followed by an expression whose value is displayed as a result of the query.
 - ☐ Assignment must always be made to a temporary relation variable.
- Example: Write r ÷ s as

$$temp1 \leftarrow \prod_{R-S} (r)$$

 $temp2 \leftarrow \prod_{R-S} ((temp1 \times s) - \prod_{R-S,S} (r))$
 $result = temp1 - temp2$

- □ The result to the right of the ← is assigned to the relation variable on the left of the ←.
- May use variable in subsequent expressions.





Find all customers who have an account from at least the "Downtown" and the Uptown" branches.

Query 1

$$\prod_{\mathsf{CN}} (\sigma_{\mathsf{BN} = \text{``Downtown''}}(depositor \bowtie account)) \cap$$

$$\prod_{CN} (\sigma_{BN=\text{"Uptown"}}(depositor \bowtie account))$$

where *CN* denotes customer-name and *BN* denotes branch-name.

Query 2

$$\Pi_{customer-name, branch-name}(depositor \bowtie account)$$

 $\div \rho_{temp(branch-name)}(\{("Downtown"), ("Uptown")\})$











Y

Find all customers who have an account at all branches located in Brooklyn city.

$$\prod_{customer-name, branch-name}$$
 (depositor \bowtie account)

$$\div \prod_{branch-name} (\sigma_{branch-city} = \text{``Brooklyn''} (branch))$$







THANK YOU!

