
DATABASE TECHNOLOGY - 1MB025

(also 1DL029, 1DL300+1DL400)

Spring 2008

An introductory course on database systems

<http://user.it.uu.se/~udbl/dbt-vt2008/>

alt. <http://www.it.uu.se/edu/course/homepage/dbastekn/vt08/>

Kjell Orsborn
Uppsala Database Laboratory
Department of Information Technology, Uppsala University,
Uppsala, Sweden

Introduction to Relational Algebra

Elmasri/Navathe ch 6
Padron-McCarthy/Risch ch 10

Kjell Orsborn

Department of Information Technology
Uppsala University, Uppsala, Sweden

Query languages

- Languages where users can express what information to retrieve from the database.
- Categories of query languages:
 - Procedural
 - Non-procedural (declarative)
- Formal (“pure”) languages:
 - Relational algebra
 - Relational calculus
 - Tuple-relational calculus
 - Domain-relational calculus
 - Formal languages form underlying basis of query languages that people use.



Relational algebra

- **Relational algebra** is a procedural language
- Operations in relational algebra takes two or more relations as arguments and return a new relation.
- Relational algebraic operations:
 - Operations from set theory:
 - Union, Intersection, Difference, Cartesian product
 - Operations specifically introduced for the relational data model:
 - Select, Project, Join
- It have been shown that the *select*, *project*, *union*, *difference*, and *cartesian product* operations form a complete set. That is any other relational algebra operation can be expressed in these.

Operations from set theory

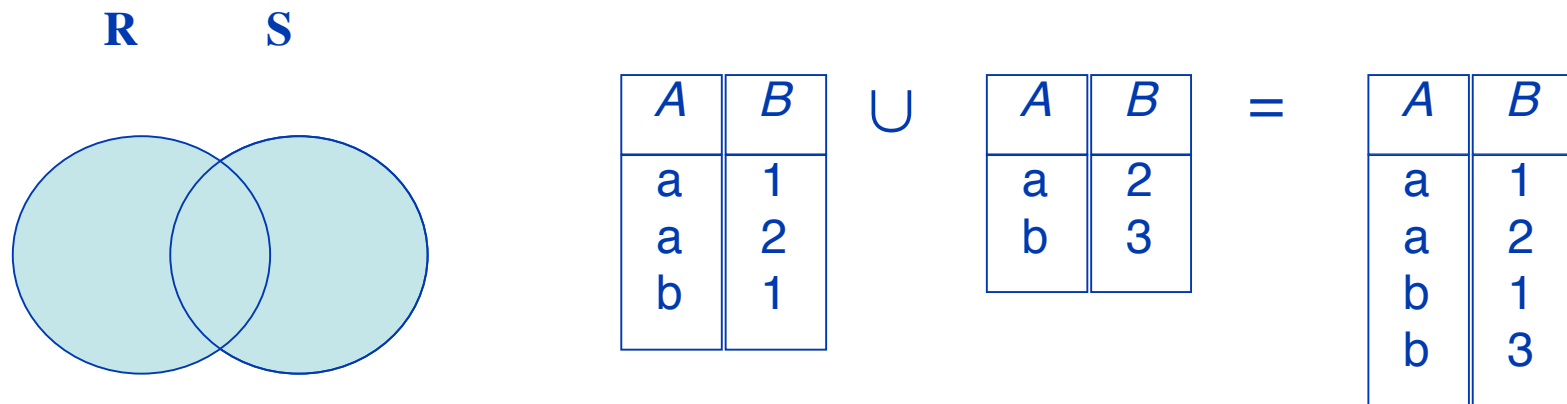
- Relations are required to be **union compatible** to be able to take part in the union, intersection and difference operations.
- Two relations R_1 and R_2 is said to be union-compatible if:

$$R_1 \subseteq D_1 \times D_2 \times \dots \times D_n \text{ and}$$
$$R_2 \subseteq D_1 \times D_2 \times \dots \times D_n$$

i.e. if they have the same degree and the same domains.

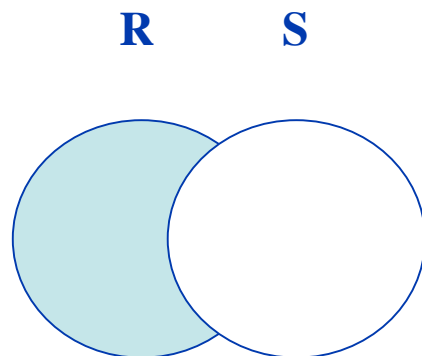
Union operation

- The **union** of two union-compatible relations R and S is the set of all tuples that either occur in R , S , or in both.
- Notation: $R \cup S$
- Defined as: $R \cup S = \{t \mid t \in R \text{ or } t \in S\}$
- For example:



Difference operation

- The **difference** between two union-compatible sets R and S is the set of all tuples that occur in R but not in S .
- Notation: $R - S$
- Defined as: $R - S = \{t \mid t \in R \text{ and } t \notin S\}$
- For example:



A	B
a	1
a	2
b	1

 -

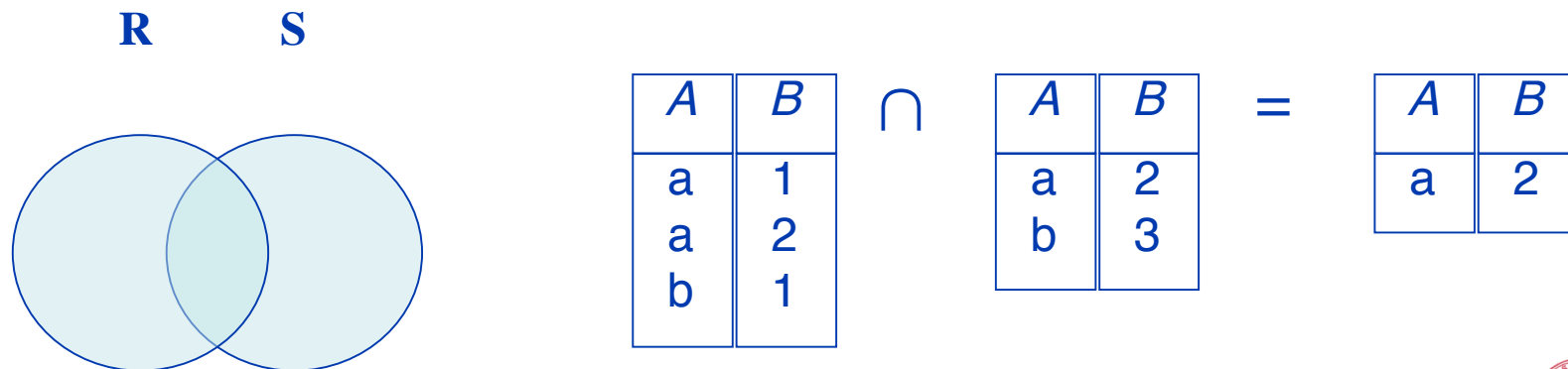
A	B
a	2
b	3

 =

A	B
a	1
b	1

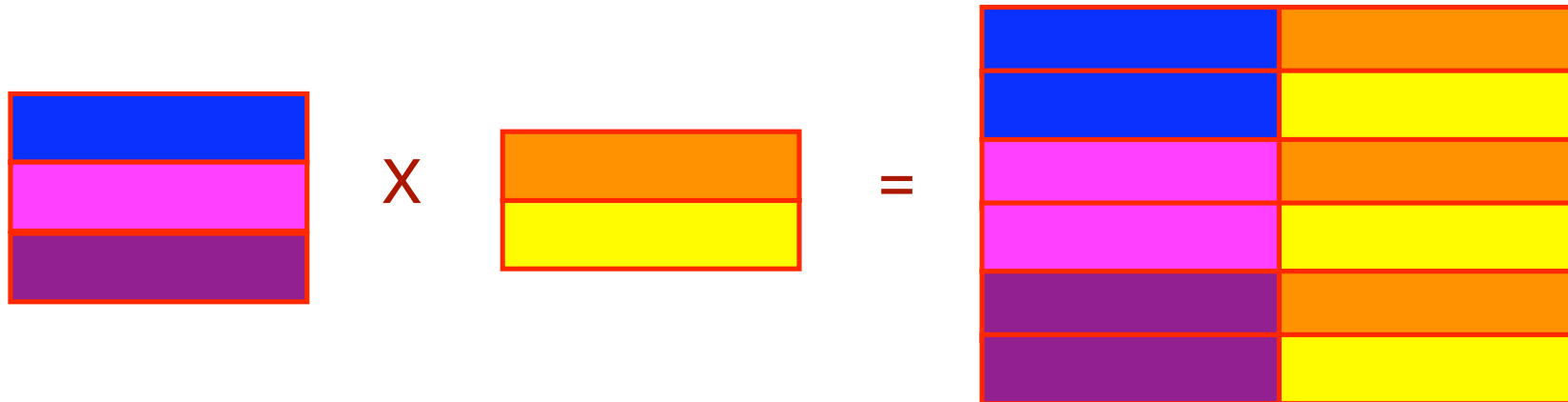
Intersection

- The **intersection** of two union-compatible sets R and S , is the set of all tuples that occur in both R and S .
- Notation: $R \cap S$
- Defined as: $R \cap S = \{t \mid t \in R \text{ and } t \in S\}$
- For example:



Cartesian product

- Let R and S be relations with k_1 and k_2 arities resp. The **cartesian product** of R and S is the set of all possible k_1+k_2 tuples where the first k_1 components constitute a tuple in R and the last k_2 components a tuple in S .
- Notation: $R \times S$
- Defined as: $R \times S = \{t \mid t \in R \text{ and } t \in S\}$
- Assume that attributes of $r(R)$ and $s(S)$ are disjoint. (i.e. $R \cap S = \emptyset$). If attributes of $r(R)$ and $s(S)$ are not disjoint, then renaming must be used.



Cartesian product example

<table border="1"><thead><tr><th><i>A</i></th><th><i>B</i></th></tr></thead><tbody><tr><td>a</td><td>1</td></tr><tr><td>b</td><td>2</td></tr></tbody></table>	<i>A</i>	<i>B</i>	a	1	b	2	×	<table border="1"><thead><tr><th><i>C</i></th><th><i>D</i></th></tr></thead><tbody><tr><td>a</td><td>5</td></tr><tr><td>b</td><td>5</td></tr><tr><td>b</td><td>6</td></tr><tr><td>c</td><td>5</td></tr></tbody></table>	<i>C</i>	<i>D</i>	a	5	b	5	b	6	c	5	=	<table border="1"><thead><tr><th><i>A</i></th><th><i>B</i></th><th><i>C</i></th><th><i>D</i></th></tr></thead><tbody><tr><td>a</td><td>1</td><td>a</td><td>5</td></tr><tr><td>a</td><td>1</td><td>b</td><td>5</td></tr><tr><td>a</td><td>1</td><td>b</td><td>6</td></tr><tr><td>a</td><td>1</td><td>c</td><td>5</td></tr><tr><td>b</td><td>2</td><td>a</td><td>5</td></tr><tr><td>b</td><td>2</td><td>b</td><td>5</td></tr><tr><td>b</td><td>2</td><td>b</td><td>6</td></tr><tr><td>b</td><td>2</td><td>c</td><td>5</td></tr></tbody></table>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	a	1	a	5	a	1	b	5	a	1	b	6	a	1	c	5	b	2	a	5	b	2	b	5	b	2	b	6	b	2	c	5
<i>A</i>	<i>B</i>																																																							
a	1																																																							
b	2																																																							
<i>C</i>	<i>D</i>																																																							
a	5																																																							
b	5																																																							
b	6																																																							
c	5																																																							
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>																																																					
a	1	a	5																																																					
a	1	b	5																																																					
a	1	b	6																																																					
a	1	c	5																																																					
b	2	a	5																																																					
b	2	b	5																																																					
b	2	b	6																																																					
b	2	c	5																																																					

Selection operation

- The selection operator, σ , selects a specific set of tuples from a relation according to a selection condition (or selection predicate) P .
- Notation: $\sigma_P(R)$
- Defined as: $\sigma_P(R) = \{t \mid t \in R \text{ and } P(t)\}$ (i.e. the set of tuples t in R that fulfills the condition P)
- Where P is a logical expression^(*) consisting of terms connected by:
 \wedge (**and**), \vee (**or**), \neg (**not**)
and each term is one of:
 <attribute> op <attribute> or <constant>
where op is one of: $=, \neq, >, \geq, <, \leq$

Example: $\sigma_{\text{SALARY} > 30000}(\text{EMPLOYEE})$

(*) a formula in propositional calculus

Selection example

$R =$

A	B	C	D
a	a	1	7
a	b	5	7
b	b	2	3
b	b	4	9

$\sigma_{A=B \wedge D > 5}(R) =$

A	B	C	D
a	a	1	7
b	b	4	9

Projection operation

- The **projection** operator, Π , picks out (or projects) listed columns from a relation and creates a new relation consisting of these columns.
- Notation: $\Pi_{A_1, A_2, \dots, A_k}(R)$
where A_1, A_2 are attribute names and R is a relation name.
- The result is a new relation of k columns.
- Duplicate rows removed from result, since relations are sets.

Example: $\Pi_{LNAME, FNAME, SALARY}(EMPLOYEE)$

Projection example

$$R =$$

A	B	C
a	1	1
a	2	1
b	3	1
b	4	2

$$\Pi_{A,C}(R) =$$

A	C
a	1
a	1
b	1
b	2

$$=$$

A	C
a	1
b	1
b	2

Join operator \bowtie

- The **join** operator, \otimes (almost, correct \bowtie), creates a new relation by joining related tuples from two relations.
- Notation: $R \otimes_C S$
 C is the join condition which has the form $A_r \theta A_s$, where θ is one of $\{=, <, >, \leq, \geq, \neq\}$. Several terms can be connected as $C_1 \wedge C_2 \wedge \dots \wedge C_k$.
- A join operation with this kind of general join condition is called “Theta join”.

Example Theta join

R		S		R $\otimes_{A \leq F} S$
----------	--	----------	--	--

<i>A</i>	<i>B</i>	<i>C</i>	$\otimes_{A \leq F}$	<i>D</i>	<i>E</i>	<i>F</i>	=	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>
1	2	3		2	3	4		1	2	3	2	3	4
6	7	8		7	3	5		1	2	3	7	3	5
9	7	8		7	8	9		1	2	3	7	8	9
6	7	8		7	8	9		6	7	8	7	8	9
9	7	8		7	8	9		9	7	8	7	8	9

Equijoin

- The same as join but it is required that attribute A_r and attribute A_s should have the same value.
- Notation: $R \otimes_C S$
 C is the join condition which has the form $A_r = A_s$. Several terms can be connected as $C_1 \wedge C_2 \wedge \dots \wedge C_k$.

Example Equijoin

$$\begin{array}{c} R \\ \begin{array}{|c|c|} \hline A & B \\ \hline a & 2 \\ \hline a & 4 \\ \hline \end{array} \otimes_{B=C} \begin{array}{c} S \\ \begin{array}{|c|c|c|} \hline C & D & E \\ \hline 2 & d & e \\ \hline 4 & d & e \\ \hline 9 & d & e \\ \hline \end{array} \end{array} = \begin{array}{c} R \otimes_{B=C} S \\ \begin{array}{|c|c|c|c|c|} \hline A & B & C & D & E \\ \hline a & 2 & 2 & d & e \\ \hline a & 4 & 4 & d & e \\ \hline \end{array} \end{array}$$

Natural join

- **Natural join** is equivalent with the application of join to R and S with the equality condition $A_r = A_s$ (i.e. an equijoin) and then removing the redundant column A_s in the result.
- Notation: $R \bowtie_{A_r, A_s} S$
 A_r, A_s are attribute pairs that should fulfil the join condition which has the form $A_r = A_s$. Several terms can be connected as $C_1 \wedge C_2 \wedge \dots \wedge C_k$.

Example Natural join

$$\begin{array}{c} R \\ \begin{array}{|c|c|} \hline A & B \\ \hline a & 2 \\ \hline a & 4 \\ \hline \end{array} \otimes_{B=C} \begin{array}{c} S \\ \begin{array}{|c|c|c|} \hline C & D & E \\ \hline 2 & d & e \\ \hline 4 & d & e \\ \hline 9 & d & e \\ \hline \end{array} \end{array} = \begin{array}{c} R *_{B=C} S \\ \begin{array}{|c|c|c|c|} \hline A & B & D & E \\ \hline a & 2 & d & e \\ \hline a & 4 & d & e \\ \hline \end{array} \end{array}$$

Composition of operations

- Expressions can be built by composing multiple operations
- Example: $\sigma_{A=C} (R \times S)$

$$R \times S = \begin{array}{|c|c|} \hline A & B \\ \hline a & 1 \\ \hline b & 2 \\ \hline \end{array} \times \begin{array}{|c|c|} \hline C & D \\ \hline a & 5 \\ \hline b & 5 \\ \hline b & 6 \\ \hline c & 5 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline A & B & C & D \\ \hline a & 1 & a & 5 \\ \hline a & 1 & b & 5 \\ \hline a & 1 & b & 6 \\ \hline a & 1 & c & 5 \\ \hline b & 2 & a & 5 \\ \hline b & 2 & b & 5 \\ \hline b & 2 & b & 6 \\ \hline b & 2 & c & 5 \\ \hline \end{array}$$

$$\sigma_{A=C} (R \times S) = \begin{array}{|c|c|c|c|} \hline A & B & C & D \\ \hline a & 1 & a & 5 \\ \hline b & 2 & b & 5 \\ \hline b & 2 & b & 6 \\ \hline \end{array}$$



Assignment operation

- The assignment operation (\leftarrow) makes it possible to assign the result of an expression to a temporary relation variable.

- Example:

$temp \leftarrow \sigma_{dno = 5}(EMPLOYEE)$

$result \leftarrow \Pi_{fname, lname, salary}(temp)$

- The result to the right of the \leftarrow is assigned to the relation variable on the left of the \leftarrow .
- The variable may be used in subsequent expressions.

Renaming relations and attribute

- The assignment operation can also be used to rename relations and attributes.
- Example:

$$\text{NEWEMP} \leftarrow \sigma_{\text{dno} = 5}(\text{EMPLOYEE})$$
$$\text{R}(\text{FIRSTNAME}, \text{LASTNAME}, \text{SALARY}) \leftarrow \\ \Pi_{\text{fname}, \text{lname}, \text{salary}}(\text{NEWEMP})$$

Division operation

- Suited to queries that include the phrase “for all”.
- Let R and S be relations on schemas R and S respectively, where

$$R = (A_1, \dots, A_m, B_1, \dots, B_n)$$
$$S = (B_1, \dots, B_n)$$

- The result of $R \div S$ is a relation on the schema $R - S = (A_1, \dots, A_m)$

$$R \div S = \{t \mid t \in \Pi_{R-S}(R) \forall u \in S \wedge tu \in R\}$$

Example Division operation

R			S		R ÷ S
A	B	÷	B	=	A
a	1		1		a
a	2		2		e
a	3				
b	1				
c	1				
d	1				
d	3				
d	4				
d	6				
e	1				
e	2				

Relation algebra as a query language

- Relational schema: $supplies(\underline{sname}, \underline{iname}, price)$
- “What is the names of the suppliers that supply cheese?”
- “What is the name and price of the items that cost less than 5 \$ and that are supplied by WALMART”

$$\pi_{sname}(\sigma_{iname='CHEESE'}(SUPPLIES))$$
$$\pi_{iname,price}(\sigma_{sname='WALMART' \wedge price < 5}(SUPPLIES))$$

Additional relational operations

- Outer join and outer union (presented together with SQL)
- Aggregation operations (presented together with SQL)
- Update operations (presented together with SQL)
 - (not part of pure query language)