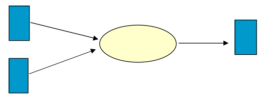
Relational algebra

- A model of how to work with the database
- What to we do with the database
  - Fetch data
  - Maintain data
- Relational Algebra specifies the operations that may be used in computing new relations from existing ones

Relational algebra

- Query:
  - Starting point: the current state of the database consists of a collection of relations
  - produces a new relation as its result.
  - is an expression that specifies how the result relation is produced from the starting relations by applying the operations of the relational algebra



Relational algebra

- Maintenance:
  - Relational algebra considers only how to construct new relations of existing ones
  - We may consider maintenance as replacing an existing relation with the result of a query

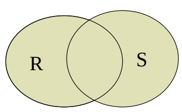
  Example:
  - Starting point: relation CAR
  - Qyery: Find out all cars but not the one with register number ABC-123. = CAR1
  - Replace the relation CAR with CAR1

Relational algebra

- Operations to build new relations from existing ones
- Basis in mathematical Set Theory
- The kernel operations of the Set Theory
  - union, difference, (cross) product, intersection
- Native operations
  - projection, select, joins

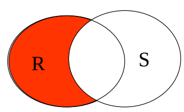
Relational algebra

- Union
  - Union builds up a relation that contains all the tuples of the operand relations.
  - $R \cup S = \{ t \mid t \in R \lor t \in S \}$,
  - R and S are relations and t is a tuple of either R or S or both. A tuple in both R and S is included in the result only once,



Relational algebra - difference

- Difference R-S extracts from relation R those tuples that do not belong to S :
- $R - S = \{ t \mid t \in R \land t \notin S \}$.



Relational algebra

- Both Union and Difference presuppose that the operands are **compatible**:
  - Same number of attributes
  - Corresponding columns share the same domains
  - Names of the corresponding columns need not be the same
  - The first operand determines the column names for the result

Relational algebra – (cross) product

- Product R × S produces a relation where each tuple of R is concatenated with each tuple of S

R	A	B
	1	2
	3	4

S	D	E
	3	4
	5	6
	1	3

RxS	A	B	D	E
	1	2	3	4
	1	2	5	6
	1	2	1	3
	3	4	3	4
	3	4	5	6
	3	4	1	3

Relational algebra - product

- **degree(R x S) = degree(R)+degree(S)**
- **cardinality(R x S) = cardinality(R)*cardinality(S)**
  - Student relation has 30 000 rows
  - Studies relation has 600 000 rows
  - Student x Studies: 18 000 000 000 rows
- If relations have columns with the same name:
  - Attach schema name as a specifier
  - R(A,B,C) x S(B,C,D) => RxS(A, R.B, R.C, S.B, S.C, D)

Relational algebra - projection

- Projection extracts from a relation the value combinations appearing in listed columns
- $\pi_{A1,\dots,An}$ (R)={(a1,…,an)| x∈R, ∀i=1..n: ai=x.Ai}
  - A1….An are attributes (column names)
  - a1,…,an are values
  - x is a tuple (row)
  - x.A denotes the value of attribute A in tuple x
- Although the same value combination a1,…,an would be included in many tuples of the starting relation it is included only once in the result = elimination of duplicates

Relational algebra - projection

T	A	B	D	E
	1	2	3	4
	1	2	5	6
	1	2	1	3
	3	4	3	4
	3	4	5	6
	3	4	1	3

π_B(T)	B
	2
	4

$\pi_{D,E}$(T)	D	E
	3	4
	5	6
	1	3

Relational algebra - selection

- Select operation extracts from a relation the rows that fulfill a given condition
- $\sigma_{condition}$(R) = { x | x∈R and *condition* is true when the attribute names in it are substituted by the values of those attributes in tuple x}
- Operand in a condition may be constraints or attributes. Standard comparison operations may be used =, ≠, <, >, <= and >=.

Relational algebra - selection

R	A	B
	1	2
	3	4

$\sigma_{A=1}(R)$	A	B
	1	2

$\sigma_{3>2}(R)$	A	B
	1	2
	3	4

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

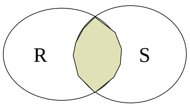
Empty relation

Relaational algebra – assignment, renaming

- Assignment renames the relation
- S(A,B,….,N) := expression.
- The result of the expressions must be compatible with the schema on the left
- Assignment is an auxiliary operation
- StudentName(Name):= $\pi_{LastNamei}$(Student)

Relational algebra - intersection

- Intersection is a set theoretic operation that picks up the common rows in two relations, each row, however, only once
- $R \cap S \equiv R-(R-S) \equiv S-(S-R)$



Relational algebra – logical expressions in conditions

- It is possible to use in complex conditions using similar logical connectives that are used in programming languages. A selection with a complex condition may be reduced to selections with simple conditions
- $\sigma_{c1 \text{ or } c2}(R) \equiv \sigma_{c1}(R) \cup \sigma_{c2}(R)$
- $\sigma_{c1 \text{ and } c2}(R) \equiv \sigma_{c1}(R) \cap \sigma_{c2}(R)$
- $\sigma_{\text{not } c1}(R) \equiv R - \sigma_{c1}(R)$

Relational algebra - join ⋈

- Concatenation of rows that satisfy a given condition
- Join combines selection and product
  - $R \,|X|_{join\_condition} S \equiv \sigma_{join\_condition}(R \times S)$
- Most common join is to join rows with rows that refer to them
  – Join condition is this case is R.K=S.FK,
  – where K is the key of R and FK is a foreign key in S that contains a reference to R

Join

R	A	B
	1	2
	3	4

S	D	E
	3	4
	5	6
	1	3

Condition: R.A=S.D

RxS	A	B	D	E
	1	2	3	4
	1	2	5	6
	1	2	1	3
	3	4	3	4
	3	4	5	6
	3	4	1	3

First product

Join

R	A	B
	1	2
	3	4

S	D	E
	3	4
	5	6
	1	3

RxS	A	B	D	E
	~~1~~	~~2~~	~~3~~	~~4~~
	~~1~~	~~2~~	~~5~~	~~6~~
	1	2	1	3
	3	4	3	4
	~~3~~	~~4~~	~~5~~	~~6~~
	~~3~~	~~4~~	~~1~~	~~3~~

Condition: R.A=S.D

Then select

Join

R	A	B
	1	2
	3	4

S	D	E
	3	4
	5	6
	1	3

RxS	A	B	D	E
	1	2	3	4
	1	2	5	6
	~~1~~	~~2~~	~~1~~	~~3~~
	~~3~~	~~4~~	~~3~~	~~4~~
	3	4	5	6
	3	4	1	3

Condition: R.A≠S.D

Natural join

- R*S
- Join condition need not be expressed, but is based on the equality of values of similarly named columns.
  - The same column name in both relations.
  - Similarly named columns are included in the result only once.

Natural join

- Let A1,…,An be attributes of R that are not included in the schema of S, and C1,…, Cm attributes of S that are not included in the schema of R. Let B1…Bk be attributes of both R and S.
- $R*S \equiv \pi_{A1,\ldots,An,R.B1,\ldots,R.Bk,C1,\ldots,Cm}$

  $(R \mid\times\mid_{R.B1=S.B1 \text{ and } \ldots \text{ and } R.Bk=S.Bk} S)$

Natural Join

- R(A,B,C) * S(A,D,E) =

  $\pi_{A,B,C,D,E} (R|X|_{R.A=S.A}\ S)$

- Q(A,B,C) * T(A,B,C) =

  $\pi_{A,B,C} (Q|X|_{Q.A=T.A \text{ and } Q.B=T.B \text{ and } Q.C=T.C}\ T)$

Outer join

- Outer join is a combination of union and join. It includes in the result also such rows that in normal join would not be included because they do not have any matching pair to satisfy the join condition (below left outer join)
  - $R \supset\!<\!|_{condition} S \equiv$

    $(R \mid\times\mid_{condition} S) \cup (R - \pi_{att(R)}(R \mid\times\mid_{condition} S))\times\aleph(S)$
- att(R) lists the attributes of R and ℵ(S) is relation with the schema of S and a single tuple each value of which is a null value.

Ulkoliitos

- Car ⊐<| $_{Car.Reg\_no=Onnership.Reg\_no}$ Ownership
  - All cars are included but if no ownership is defined for some car the Ownership attributes in the concatenated tuple are null values.

Outer join

Auto	Reknro	…
	ABC-123	
	DEF-456	
	GHI-789	

Omistus	Reknro	Omistaja
	ABC-123	Liisa
	ABC-123	Kalle
	GIH-789	Pekka

Tulos	A.Reknro	O.Reknro	Omistaja
	ABC-123	ABC-123	Liisa
	ABC-123	ABC-123	Kalle
	GIH-789	GIH-789	Pekka
	DEF-456	NULL	NULL