Relational Algebra and Relational Calculus :
Theoretical Basis of Database

Relational Algebra **[**Basis for SQL]

Relational Algebra Operations

Set operations:

Union	U	(all in either)	+
Intersection	∩	(all in both)	*
Set difference	-	(in first but not second)	
Cartesian product (a.k.a. Cross Product)	**X**	(combine every tuple in A with every tuple in B)	

Unary operations:

Select σ
(Keep tuples satisfy condition;
QBE uses criteria)

Project π
(Keep attributes desired;
QBE uses check marks)

Rename ρ
(like assignment statement)

Relational Algebra Binary Operations

Join |**X**|

(combine tuples from different relations)
(like Cartesian product with select)

equijoin
Natural join
(leaves out 2nd identical attribute)

Inner join
(tuples with no match dropped)

Outer join
(keep all tuples)

Left (outer) join
(keep tuples in first/left table)

Right (outer) join
(keep tuples in second/left table)
Division
Outer union
(full outer join on compatible attributes)

Aggregate functions
Sum
Average
Maximum
Minimum
Count

Assignment Statements
<set2> <= <set1>

Relational Algebra:
Set operations: union, intersection, set difference, Cartesian product(cross product).
Operations for relational databases: select, project, join.
Additional aggregate functions: summaries.
Logical functions: AND, OR, NOT

NOTE: Select in R.A. is NOT Select in SQL.
SELECT: (vertical partition: choosing tuples/rows)
σ <condition>(R)
Select from relation based on CONDITION.
σ <deptID=4>(employeeTable)
CONDITION can use AND, OR, NOT, =,<,> constants

SELECT s applied to each tuple individually in ONE relation.
In general, if <condition> is true, the TUPLE is included.
SELECT is commutative.
[SQL: WHERE]

PROJECT: (horizontal partition: choosing attributes/columns)
π <attributes.>(R)
In general, the attributes of interest in R (in ALL tuples) are included.
[like Select in SQL]

PROJECT removes duplicates. [SQL does not always remove duplicates]
[More like SQL SELECT DISTINCT attributes FROM relationship]

You can apply multiple relational algebra operations in a sequence:
π <name>(σ <deptID=5> EMPLOYEE))
OR can use assignment statements:
DEPT5 <= (σ <deptID=5> EMPLOYEE)
RESULT<=π <name>DEPT5

Can also RENAME relation names or attribute names, ρ.

If you use consistent names, RENAME is not needed often.

Relational Algebra, continued

UNION, INTERSECTION, SET DIFFERENCE (minus, except)
They can only be applied to tuple compatible relations; (same attributes).

UNION: U +

ALL tuples in EITHER of the two sets

INTERSECTION ∩ *

All tuples in BOTH of the two sets

DIFFERENCE: -

All tuples in the first set but not the second set

Cartesian /Cross Product **X**
Binary operation, but relations do NOT need to be compatible.

Rarely used by itself, except by accident.
Most common: SELECT tuples of interest after a cross product.

Cross product of all employees with all dependents,
Select ones where employeeID == familyEmployee
EMP-Depend <= EMP X DEPEND
Actual<= σ <eID=Femp>(EMP-Depend)

JOIN |X|
Can be done with cross product followed by selection.
VERY frequent in Database queries.
R |X| <join condition> S

EMP |X| <eID=Femp>DEPEND

Relational Algebra, continued

The join condition is called a THETA.
A join with a general condition is called a THETA JOIN.

Some TUPLES do NOT appear in the result (for NULLs or not matching condition.)
Therefore, JOIN does NOT preserve all information.

EQUIJOIN: attributes that have identical values.
NATURAL JOIN gets rid of the (surplus) identical value.
[Match each attribute that has the same name]
The attribute over which a JOIN takes place is called the JOIN Attribute.
If have the same attribute name, renaming is not necessary.

You should be familiar with:
Natural Join, Left Join, Right Join, INNER join

SELECT, PROJECT, UNION, RENAME, MINUS, CROSS are a complete set.
Intersection can be expressed in terms of UNION and MINUS.
Join can be expressed by CROSS and SELECT.

DIVISION operation: : (For Every)
Names of people on ALL of John's projects.
(See Relational Division in SQL)

R(Z) / S(X) [attributes of S a subset of attributes of R]
RESULT is all tuples of Z that match EVERY TUPLE in X.
Used for "ALL" queries.

Star Trek: List roles that occur in EVERY/ALL series.
ROLES / SERIES

Registration: List courses that ALL CIS programs require.
COURSES / PROGRAM

Generalize Projection with Renaming for computed fields

Aggregate functions:
Count, Max, Min, Sum, Average

Recursive Closure
BOSS (one person supervises another)
Proposed transitive closure of relations

OUTER JOINS

SUPERVISE (sID, deptNUM)

List all employees who supervise a department
Departments <= PROJECT<deptNUM>(DEPT)
Super <= PROJECT<sID,sName> SUPERVISE |X| <deptNUM==deptNUM> Departments

Example queries, 265ff.

Remember:
Select chooses TUPLES (rows)
Project chooses attributes (columns)

Discipline number section
SELECT <discipline==CIS>
PROJECT discipline number

Even if you pull from a SECTION table,
It will only have ONE row/tuple for each CIS course
RENAME
Student in StudentTable
Sid in advising table
Rename Sid As StudentID
RenameStudent As StudentID
sID   NamePerson Address Phone

fID   NamePerson Address Phone
UNION all faculty AND all students
INTERSECTION all faculty who are also students
DIFFERENCE
All faculty who are NOT students
Q1= PROJECT(SELECT(R1))
Q2= PROJECT(SELECT(R2))
All= Q1 + Q2
Few = Q1 * Q2

Cross Product: of AB with CDE
A1 B1
A2 B2
A3 B3

C1 D1 E1
C2 D2 E2
Each tuple matches with EVERY tuple in the other table
A1 B1 C1 D1 E1
A2 B2 C1 D1 E1
A3 B3 C1 D1 E1
A1 B1 C2 D2 E2
A2 B2 C2 D2 E2
A3 B3 C2 D2 E2

3*2=6 items in the cross product table
SELECT is often to match attributes

Relational Algebra Summary

Set operations:

Union
Intersection
Set difference
Cartesian product
(a.k.a. Cross Product)

Unary operations:

Select
(Keep tuples satisfy condition;
QBE uses criteria)
Project
(Keep attributes desired;
QBE uses check marks)
Rename
(like assignment statement)

Binary Operations:

Set Theory
Union
(all in either)
Intersection
(all in both)
Minus
(in first but not second)

Cross Product
(combine every tuple in A with every tuple in B)

Relational operations

Join
(combine tuples from different relations)
(like Cartesian product with select)
equijoin
Natural join
(leaves out 2nd identical attribute)
Inner join
(tuples with no match dropped)
Outer join
(keep all tuples)
Left (outer) join
(keep tuples in first/left table)
Right (outer) join
(keep tuples in second/left table)
Division
Outer union
(full outer join on compatible attributes)

Aggregate functions

Sum
Average
Maximum
Minimum
Count