
Chapter# 9

Domains & Relational Algebra

Domains

- The primary purpose of domains in SQL is to allow a built-in type to be given a name.
 - Create Domain Dealer CHAR (5);
 - Create Domain Car CHAR (6);
 - Create Table D (D# Dealer ,);
 - Create Table C (C# Car , ...);

Differences between True Domains and SQL Domains

- SQL domains are just a syntactic shorthand
- Values in SQL domains can not be of arbitrary internal complexity. They are limited to the complexity of the built-in types.
- An SQL domain has to be defined in terms of exactly one built-in type

-
- An SQL domain can not have more than one possible representation
 - There is no strong typing in SQL. The following SQL operation will not fail on a type checking:
Select *
From DC
Where D# = C#;

-
- SQL supports exactly eight true relational domains:
 - numbers
 - character strings
 - bit strings
 - dates
 - times
 - timestamps
 - year/month intervals
 - day/time intervals

-
- SQL does not allow users to define the operators that apply to a given domain
 - The operators are those built-in operators that apply to the corresponding representation.

Syntax

- Create domain <domain name> <built-in type name> [<default spec>] [<constraints>];
- <default spec> specifies a default value that applies to every column in case there is no explicit value
- <constraints> specify certain integrity constraints that apply to the domain

-
- An SQL domain can be altered at any time in a variety of ways by means of Alter Domain statement
 - The alter domain allows a default value to be changed or deleted. Also it allows an integrity constraints to be defined or deleted.

-
- An SQL domain can be dropped
Drop Domain <domain name><option>
 - <option> is either Restrict or Cascade
 - Drop will fail if the domain is referenced anywhere and Restrict is used
 - If Cascade is used the columns defined using the dropped domain will be defined using the underlying built-in domain

Relational Algebra

- Relational algebra: a set of operators that take relations as their operands and return a relation as their result.
- Relational algebra was introduced by Codd in 1972
- Codd has introduced eight operators
- It has evolved considerably since that time

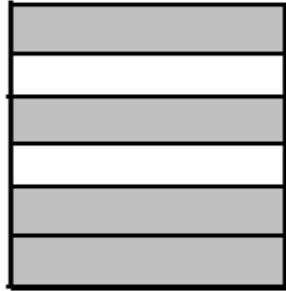
Codd's original operators

- Restrict
- Project
- Product
- Union
- Intersection
- Difference
- Join
- Divide

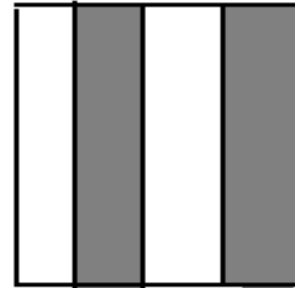
-
- Restrict: returns a relation containing all tuples from a specified relation that satisfy a specified condition
 - Project: returns a relation containing all sub(tuples) that remain in a specified relation after specified attributes have been removed
 - Product: returns a relation containing all possible tuples that are a combination of two tuples, one from each of two specified relations.

-
- **Union:** Returns a relation containing all tuples that appear in either or both of two specified relations.
 - **Intersect:** returns a relation containing all tuples that appear in both of the two specified relations.
 - **Difference:** returns a relation containing all tuples that appear in the first and not the second of two specified relations.

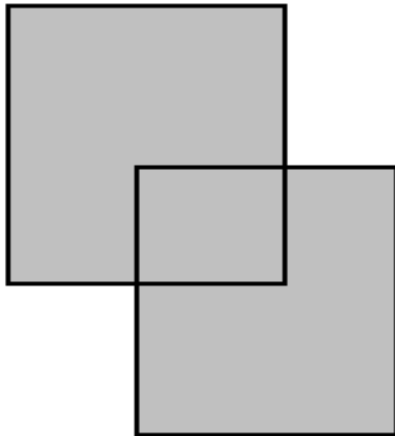
Restrict



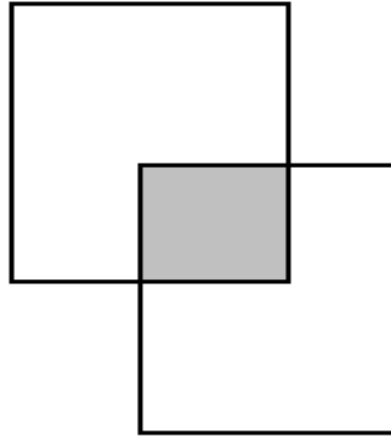
Project



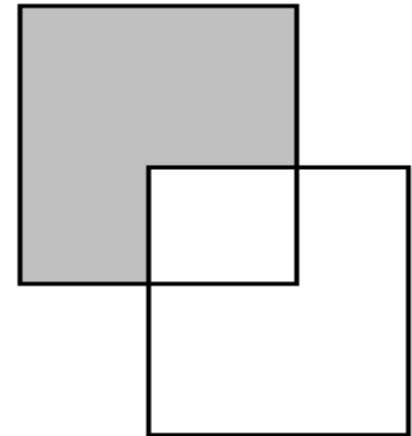
Union



Intersection



Difference



Product

a
b

x
y
z

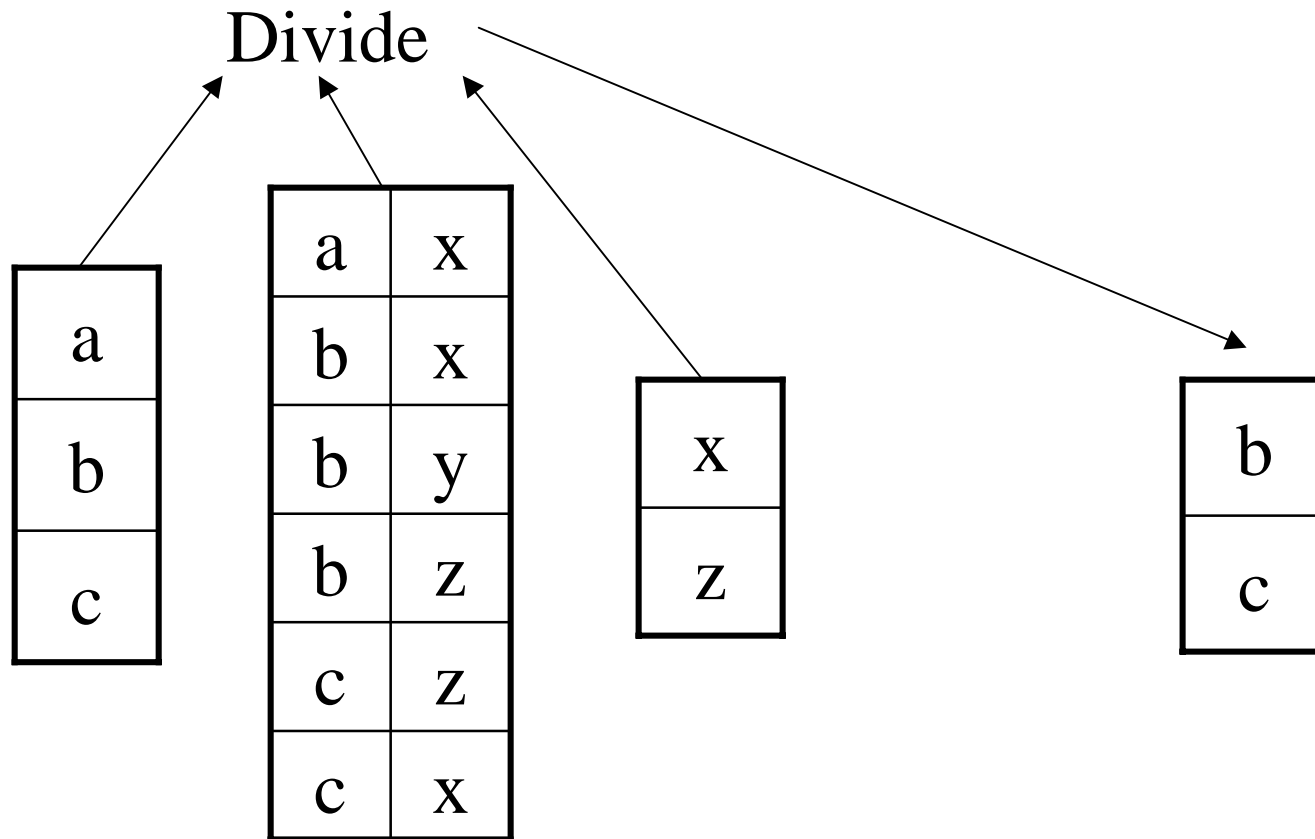
a	x
a	y
a	z
b	x
b	y
b	z

(Natural) Join

x1	y1
x1	y2
x2	y3

y1	z2
y2	z1
y3	z1

x1	z2
x1	z1
x2	z1



Syntax

- Most database books use mathematical symbols for relational operators
- Date's book uses key words such as join, times, where,

Examples of notion not used in Date's book

- $R1 = \sigma_C(R2)$! Selection
where C is a condition involving the attributes of
relation R2.
- $R1 = \pi_L(R2)$! Projection
where L is a list of attributes from the schema of
R2.

Tables to demonstrate UNION and MINUS operations

Let Table A be

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT
D3	THOMAS	40	SHELTON

Let Table B be

B:

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT
D4	RAJ	30	STAMFORD

Union

- Mathematical union of two sets contains all elements belonging to either or both
- The result of the mathematical union might not be a relation
- Relational union of two sets: relations must be of the same type to preserve the closure property

A UNION B:

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT
D3	THOMAS	40	SHELTON
D4	RAJ	30	STAMFORD

Relational Intersect & Difference

- Relational intersection and difference are defined for relations of the same type as the case of union

A INTERSECT B:

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT

A MINUS B:

D#	DNAME	STATUS	LOCATION
D3	THOMAS	40	SHELTON

B MINUS A:

D#	DNAME	STATUS	LOCATION
D4	RAJ	30	STAMFORD

Product

- Relational Cartesian product contains tuples formed by the union of two tuples, the first tuple comes from the first relation and the second tuple comes from the second relation

A:

D#
D1
D2
D3
D4

B:

C#
C1
C2
C3

**CARTESIAN
PRODUCT
OF A and B**

D#	C#
D1	C1
D1	C2
D1	C3
D2	C1
D2	C2
D2	C3
D3	C1
D3	C2
D3	C3
D4	C1
D4	C2
D4	C3

Restrict

- The restriction operator yields a horizontal subset of the given relation for which some specified restriction condition is specified.
- Restriction is defined only on a single condition
 - $A \text{ Where } C1 \text{ and } C2 = (A \text{ where } C1) \text{ Intersect } (A \text{ Where } C2)$
 - $A \text{ Where } C1 \text{ Or } C2 = (A \text{ Where } C1) \text{ Union } (A \text{ Where } C2)$

Example of **restrict** operation:

D WHERE LOCATION = 'SHELTON';

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D3	THOMAS	40	SHELTON
D3	SMITH	50	SHELTON

D WHERE DNAME = 'JOHN' OR LOCATION = 'BRIDGEPORT';

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT

Project

- The projection operator yields a vertical subset of a given relation.
- The subset is obtained by removing all attributes not mentioned in the specified comma-list of attribute names then eliminating duplicate subtuples from what is left.

{ C#,MODEL }

C#	MODEL
C1	VOLVO850
C2	TAURUS
C3	FORD
C4	HONDA
C5	BMW
C6	SATURN
C7	BMW
C8	FORD

C WHERE MODEL =
'FORD' {C#,LOCATION};

C#	LOCATION
C3	STAMFORD
C8	SHELTON

D_X

D#	DNAME	STATUS	LOCATION
D1	JOHN	40	SHELTON
D2	MIKEL	20	BRIDGEPORT
D3	RON	30	STAMFORD

C_X

C#	MODEL	YEAR	LOCATION
C1	VOLVO850	1990	SHELTON
C2	TAURUS	1996	WATERBURY
C3	FORD	1989	STAMFORD
C4	HONDA	1993	SHELTON

Join

- Natural join: $D_X \text{ Join } C_X$
(EQUI JOIN OF D_X and C_X)
RESULT OF EQUI JOIN:

D#	DNAME	STATUS	LOCATION	C#	MODEL	YEAR
D1	JOHN	40	SHELTON	C1	VOLVO850	1990
D2	JOHN	40	SHELTON	C4	HONDA	1993
D3	RON	30	STAMFORD	C4	HONDA	1993

- Θ Join: Join two relations together on the basis of some comparison operator other than equality
 - (A Times B) Where $X \Theta Y$
 - GREATER THAN JOIN:

D#	DNAME	STATUS	LOCATION	C#	MODEL	YEAR	LOCATION
D2	MIKEL	20	BRIDGEPORT	C1	VOLVO850	1990	SHELTON
D2	MIKEL	20	BRIDGEPORT	C2	TAURUS	1996	WATERBURY
D2	MIKEL	20	BRIDGEPORT	C3	FORD	1989	STAMFORD
D2	MIKEL	20	BRIDGEPORT	C4	HONDA	1993	SHELTON

Divide

➤ Definition

- A has headings { X1 , X2 ,, Xm }
- B has headings { Y1 , Y2 ,, Ym }
- C has a heading that is the union of A & B headings
- A DIVIDEBY B Per C
 - The results consists of of those X values from A whose corresponding Y values in C include all Y values from B

Table1

C#
C1
C2
C3
C4

Table 2

C#	D#
C1	D1
C1	D2
C1	D3
C1	D4
C1	D5
C2	D2
C2	D3
C3	D1
C4	D1
C4	D3
C4	D4

A1:

D#
D1

A2:

D#
D1
D2
D3

**Table1 Divided by A1
per Table2**

C#
C1
C3
C4

**Table1 Divided by A2
per Table2**

C#
C1
C2

Associative and Commutative

- Union, Intersect, Times and Join are associative
 - $A \cup B \cup C = (A \cup B) \cup C$
- Union Intersect, Times and Join are commutative
 - $A \cup B = B \cup A$

Operators

σ Selection

ρ Renaming

π Projection

\bowtie Natural join

\bowtie_{θ} Theta-join

\cup Union

\cap Intersection

- Difference

Operator Precedence

- The normal way to group operators is:
 - Unary operators σ , ρ , π and \neg have highest precedence.
 - Next highest are the "multiplicative" operators, $| \times |$, $| \times |_{\theta}$, and \times .
 - Lowest are the "additive" operators, \cup , \cap , and $-$.

-
- There is no universal agreement, so we always put parentheses around the argument of a unary operator, and it is a good idea to group all binary operators with parentheses enclosing their arguments.
 - Example
 - Group $R \cup \sigma S \mid \times \mid T$ as $R \cup (\sigma(S) \mid \times \mid T)$.