Principles of Data Management

Alvin Lin

August 2018 - December 2018

Relations and Relational Algebra

id	name	DOB	height
1	Jane	12/1	5'8"
2	Joe	7/13	null

In this table, the rows (tuples) and columns (attributes) contain data and type information about the data. A database domain defines the attributes of the database. It is atomic and every domain can have a null value.

Suppose $A_1, A_2, \ldots, A_n$ are attributes. A relation R is a subset of domains. For example:

$$Dog = (breed, color, goodboy?, name, gender, weight, DOB, owner, address)$$

Relation items are unordered. An instance of a relation is defined by a table and an element of the instance is a row.

Keys

Keys are defined as a subset of attributes from a relation. The following keys are all single relation keys.

- super key: any combination of attributes that uniquely identify a row.
- candidate key: minimal super key
- primary key: one of the candidate keys

Foreign keys allow for referencing relations. A value for a given foreign key attribute must appear in another table.

Relational Algebra

- Select Operation (σ): if we have a table R with the following attributes:

A	B	C	D
x	x	1	7
x	y	5	7
y	y	13	5
y	y	30	4

If we select on this relation using the condition A = B (denoted $\sigma_{A=B}(R)$), we get the last two rows.

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

A	B	C	D
x	x	1	7
y	y	13	5
y	y	30	4

Any number of conditions can be used in a select operation.

- Projection (Π): limits the number of columns selected

A	B	C
x	1	0
y	2	1
x	3	0
y	4	2

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

A	C
x	0
y	1
x	0
y	2

Because this has duplicates, this would reduce to:

A	C
x	0
y	1
y	2

- Union: merging two relations that have the same attributes

$R =$

A	B
x	1
y	2
z	3

$S =$

A	B
m	5
n	6

$T = \sigma_{B>5}(R \cup S) =$

A	B
x	1
y	2
z	3
m	5
n	6

- Set Difference: all tuples in R not in S

$R =$

A	B
a	1
a	2
b	1

$S =$

A	B
a	2
a	3

$R - S =$

A	B
a	1
b	1

- Set Intersection: set of all tuples in both R and S, can also be rewritten using set difference.

$$R \cup S = R - (R - S)$$

- Cross/Cartesian Product: maps everything in R to everything in S.

$R =$

A	B
a	1
b	1

$S =$

C	D	E
x	a	1
y	c	5
z	f	7

$R \times S =$

A	B	C	D	E
a	1	x	a	1
a	1	y	c	5
a	1	z	f	7
b	z	x	a	1
b	z	y	c	5
b	z	z	f	7

- Renaming: not really a real operation, but allows us to resolve attribute name conflicts.

$R =$

A	B
1	2
3	4

$R \times P_S(R) =$

RA	RB	SA	SB
1	2	1	2
1	2	3	4
3	4	1	2
3	4	3	4

- Natural Join: given two relations R and S, where R and S have attributes with the same name for every tuple, if the values of those attributes have the same name, join them.

$R =$

A	B	C	D
a	1	a	a
b	2	c	a
c	4	b	b
a	1	c	a
d	2	b	b

$S =$

B	D	G
1	a	a
3	a	b
1	a	c
2	b	d
3	b	e

$R \bowtie S =$

A	B	C	D	E
a	1	a	a	a
a	1	a	a	c
a	1	c	a	a
a	1	c	a	c
d	2	b	b	d

Note that this is logically equivalent to:

$$\Pi_{A,R.B,C,R.D,E}(\sigma_{R.B=S.B \wedge R.D=S.D}(R \times S))$$

but it cannot be generalized like set intersection because one needs to know the common attributes of both tables.

Relational algebra is not Turing-complete because it has no persistent storage, and each statement always terminates with the output of a table.

Example

Get the names and ages of all dogs who weight more than 15 pounds:

$$\Pi_{name,age}(\sigma_{weight>15}(Dog))$$

or in relational calculus:

$$\{P \mid \exists D \in Dog(D.weight > 15 \wedge P.name = D.name \wedge P.age = D.age)\}$$

You can find all my notes at `http://omgimanerd.tech/notes`. If you have any questions, comments, or concerns, please contact me at alvin@omgimanerd.tech