>>

Relational Algebra

^ >>

❖ Relational Algebra

Relational algebra (RA) can be viewed as ...

- mathematical system for manipulating relations, or
- data manipulation language (DML) for the relational model

Relational algebra consists of:

- operands: relations, or variables representing relations
- operators that map relations to relations
- rules for combining operands/operators into expressions
- rules for evaluating such expressions

Why is it important?

- because it forms the basis for DBMS implementation
- relational algebra ops are like the machine code for DBMSs

❖ Relational Algebra (cont)

Core relational algebra operations:

- rename: change names of relations/attributes
- selection: choosing a subset of tuples/rows
- projection: choosing a subset of attributes/columns
- union, intersection, difference: combining relations
- product, join: combining relations

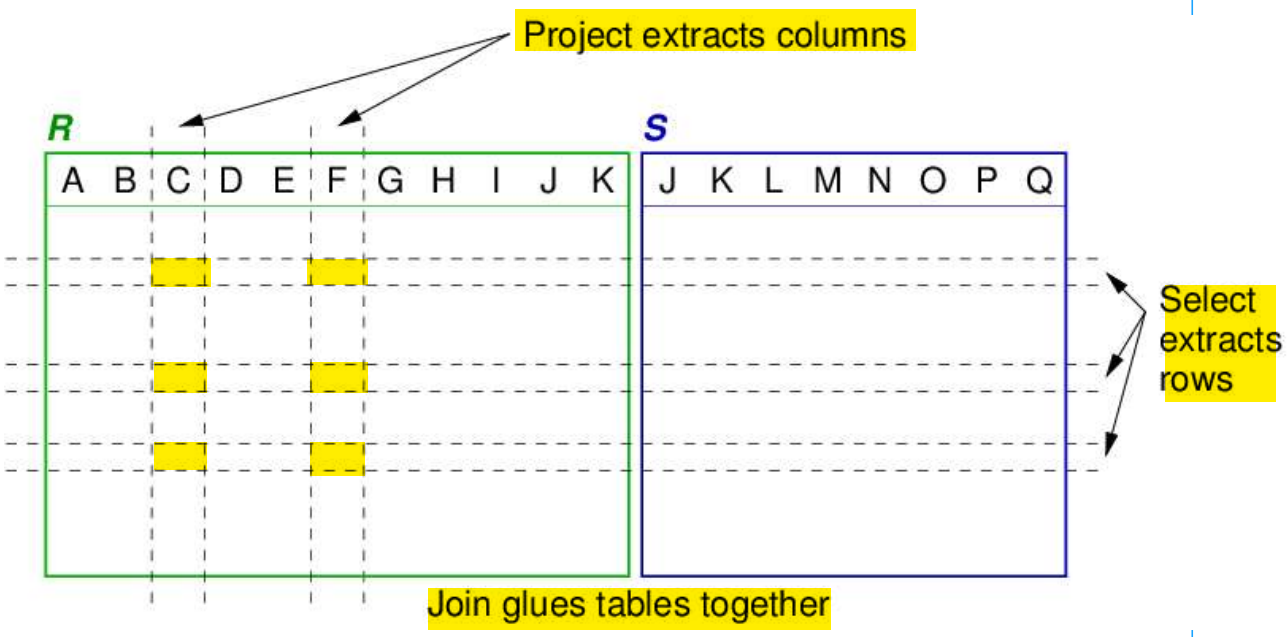
Common extensions include:

- aggregation, projection++, division

<< ^ >>

❖ Relational Algebra (cont)

Select, project, join provide a powerful set of operations for building relations and extracting interesting data from them.



Adding set operations and renaming makes RA complete.

<< ^ >>

❖ Notation

Standard treatments of relational algebra use Greek symbols.

We use the following notation (because it is easier to reproduce):

Operation	Standard Notation	Our Notation
Selection	$\sigma_{expr}(Rel)$	*Sel[expr](Rel)*
Projection	$\pi_{A,B,C}(Rel)$	*Proj[A,B,C](Rel)*
Join	$Rel_1 \bowtie_{expr} Rel_2$	Rel_1 *Join[expr]* Rel_2
Rename	$\rho_{schema} Rel$	*Rename[schema](Rel)*

For other operations (e.g. set operations) we adopt the standard notation.
Except when typing in a text file, where ***** = intersection, **+** = union

<< ^ >>

❖ Describing RA Operations

We define the semantics of RA operations using

- "conditional set" expressions e.g. *{ X | condition on X }*
- tuple notations:
  - *t[AB]* (extracts attributes *A* and *B* from tuple *t*)
  - *(x,y,z)* (enumerated tuples; specify attribute values)
- quantifiers, set operations, boolean operators

For each operation, we also describe it operationally:

- give an algorithm to compute the result, tuple-by-tuple
- the algorithm is not generally how it will be computed in practice

<< ^ >>

❖ Describing RA Operations (cont)

All RA operators return a result of type relation.

For convenience, we can name a result and use it later.

E.g.

```
Temp = R op1 S op2 T
Res  = Temp op3 Z
-- which is equivalent to
Res  = (R op1 S op2 T) op3 Z
```

Each "intermediate result" has a well-defined schema.

<< ^ >>

❖ Example Database #1

R

A	B	C	D
a	1	x	4
b	2	y	5
c	4	z	4
d	8	x	5
e	1	y	4
f	2	x	5

S

D	E	F
1	a	x
2	b	y
3	c	x
4	a	y
5	b	x

<< ^ >>

❖ Example Database #2

Beers(name,manf)
(VB, Carlton) (New, Tooheys) (Porter, Maltshovel) ...

Beers(name,addr,licence)
(CBH, Coogee,433122) (Royal, Randwick, 632987) (Regent, Kingsford,112112 ...

Frequents(drinker,bar)
(John, CBH) (Gernot, CBH) (Gernot, Regent) ...

Likes(drinker, beer)
(Andrew, New) (Gernot, Porter) (John, Pale Ale) ...

Beers(name,addr,phone)
(John, Alexandria, 93111139 (Gernot, Newtown, 92422429) (Andrew, Glebe, 90411049) ...

Sells(beer,bar,price)
(CBH, New, 2.50) (CBH, VB, 1.99 Royal, Porter, 3.00 ...

<< ^ >>

❖ Rename

Rename provides "schema mapping".

If expression E returns a relation $R(A_1, A_2, \ldots A_n)$, then

$Rename[S(B_1, B_2, \ldots B_n)](E)$

gives a relation called S

- containing the same set of tuples as E
- but with the name of each attribute changed from A_i to B_i

Rename is like the identity function on the *contents* of a relation

The only thing that Rename changes is the schema.

<< ^ >>

❖ Rename (cont)

Rename can be viewed as a "technical" apparatus of RA.

We can also use implicit rename/project in sequences of RA operations, e.g.

dirty one-liner

```
--  R(a,b,c),  S(c,d)
Res = Rename[Res(b,c,d)](Project[b,c](Sel[a>5](R)) Join S)
-- vs
Tmp1 = Select[a>5](R)
Tmp2 = Project[b,c](Tmp1)
Tmp3 = Rename[Tmp3(cc,d)](S)
Tmp4 = Tmp2 Join[c=cc] Tmp3
Res  = Rename[Res(b,c,d)](Tmp4)
-- vs
Tmp1(b,c)  = Select[a>5](R)
Tmp2(cc,d) = S
Res(b,c,d) = Tmp1 Join[c=cc] Tmp2
```

explicit renaming

implicit renaming on the LHS assignment

I like this one the most!

In SQL, we achieve a similar effect by defining a set of views

<< ^ >>

❖ Selection

Selection returns a subset of the tuples in a relation $r(R)$ that satisfy a specified condition C.

$\sigma_C(r) = Sel[C](r) = \{ t \mid t \in r \wedge C(t) \}$

C is a boolean expression on attributes in R.

Result size: $|\sigma_C(r)| \leq |r|$

Result schema: same as the schema of r (i.e. R)

Algorithmic view:

```
result = {}
for each tuple t in relation r
    if (C(t)) { result = result ∪ {t} }
```

<< ^ >>

❖ Selection (cont)

Examples of selection:

R

A	B	C	D
a	1	x	4
b	2	y	5
c	4	z	4
d	8	x	5
e	1	y	4
f	2	x	5

Sel[A=C](R)

A	B	C	D

Sel[B=1](R)

A	B	C	D
a	1	x	4
e	1	y	4

Sel[B>=D](R)

A	B	C	D
c	4	z	4
d	8	x	5

Sel[A=b or A=c](R)

A	B	C	D
b	2	y	5
c	4	z	4

<< ^ >>

❖ Selection (cont)

Querying with relational algebra (selection) ...

- Details of all bars in The Rocks

```
Result = Sel[addr=The Rocks](Bars)
```

- Beers made by Sierra Nevada

```
SNBeers = Sel[manf=Sierra Nevada](Beers)
Result  = Rename[beer](Proj[name](SNBeers))
```

<< ^ >>

❖ Projection

Projection returns a set of tuples containing a subset of the attributes in the original relation.

$\Pi_X(r)$ = *Proj[X](r)* = $\{ t[X] \mid t \in r \}$, where *r(R)*

X specifies a subset of the attributes of *R*.

Note that removing key attributes can produce duplicates.

In RA, duplicates are removed from the result set.
(In RDBMS's, duplicates are retained (i.e. they use bags, not sets))

Result size: $|\Pi_X(r)| \leq |r|$ Result schema: *R'(X)*

Algorithmic view:

```
result = {}
for each tuple t in relation r
    result = result ∪ {t[X]}
```

<< ^ >>

❖ Projection (cont)

Examples of projection:

R

A	B	C	D
a	1	x	4
b	2	y	5
c	4	z	4
d	8	x	5
e	1	y	4
f	2	x	5

Proj[A,B,C](R)

A	B	C
a	1	x
b	2	y
c	4	z
d	8	x
e	1	y
f	2	x

Proj[B,D](R)

B	D
1	4
2	5
4	4
8	5

Proj[D](R)

D
4
5

<< ^

❖ Projection (cont)

Querying with relational algebra (projection)...

- Names of all beers

```
Result = Proj[name](Beers)
```

- Names of drinkers who live in Newtown

```
Result = Proj[name](Sel[addr=Newtown](Drinkers))
```

- What are all of the breweries?

```
Result(brewer) = Proj[manf](Beers)
```

Produced: 6 Nov 2020