Relational Database History

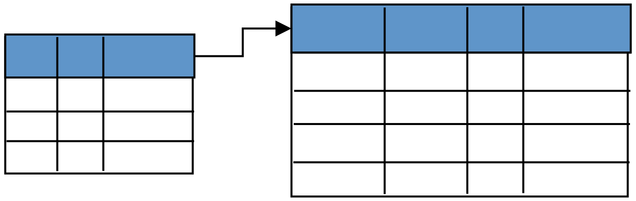
Pre-Relational: if your data changed, your application broke.

Early RDBMS were buggy and slow (and often reviled), but required only 5% of the application code.

"Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed." -- *Codd 1979*

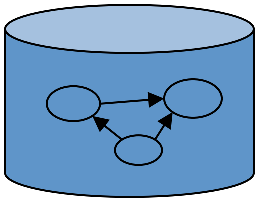
Key Ideas: Programs that manipulate tabular data exhibit an algebraic structure allowing reasoning and manipulation independently of physical data representation

Key Idea: “Physical Data Independence”



relations

physical data independence

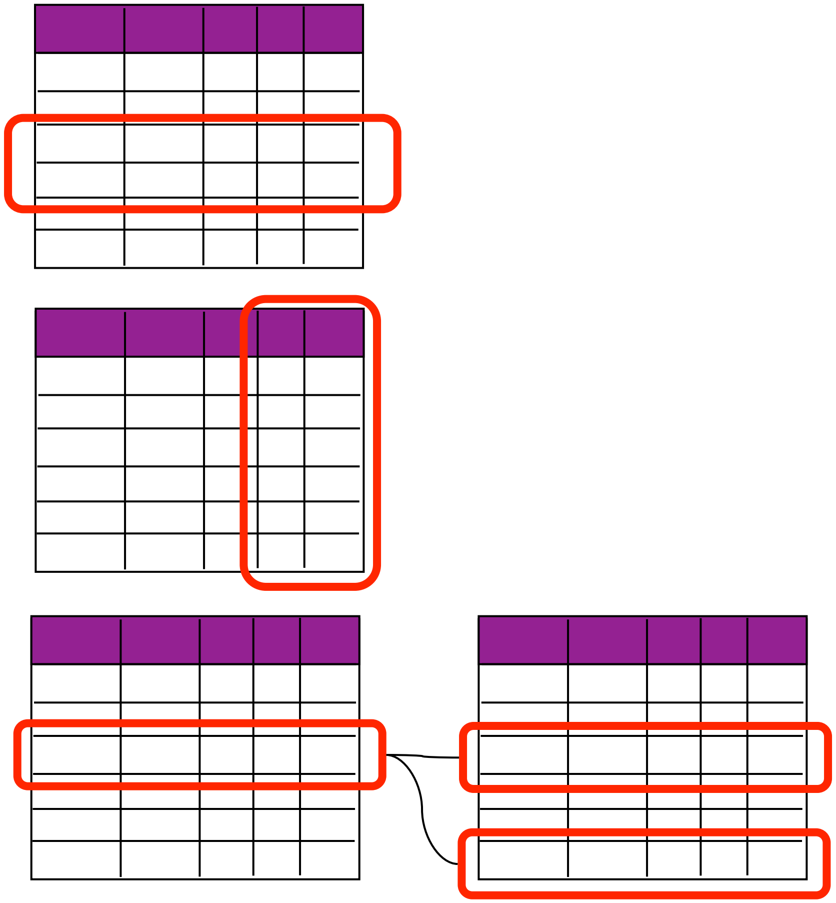


files and pointers

```
 SELECT seq
   FROM ncbi_sequences
  WHERE seq = ‘GATTACGATATTA’;
```

```
f = fopen(‘table_file’);
fseek(10030440);
while (True) {
  fread(&buf, 1, 8192, f);
  if (buf == GATTACGATATTA) {
    . . .
```

Key Idea: An *Algebra of Tables*



select

project

join

Other operators: aggregate, union, difference, cross product

Key Idea: Algebraic Optimization

$N = ((z*2)+((z*3)+0))/1$

Algebraic Laws:

1. (+) identity: $x+0 = x$
2. (/) identity: $x/1 = x$
3. (*) distributes: $(n*x+n*y) = n*(x+y)$
4. (*) commutes: $x*y = y*x$

Apply rules 1, 3, 4, 2:

$N = (2+3)*z$

two operations instead of five, no division operator

Same idea works with the Relational Algebra!

Equivalent logical expressions; different costs

$\sigma_{p=knows}(R) \bowtie_{o=s} (\sigma_{p=holdsAccount}(R) \bowtie_{o=s} \sigma_{p=accountHomepage}(R))$

right associative

$(\sigma_{p=knows}(R) \bowtie_{o=s} \sigma_{p=holdsAccount}(R)) \bowtie_{o=s} \sigma_{p=accountHomepage}(R)$

left associative

$\sigma_{p1=knows\ \&\ p2=holdsAccount\ \&\ p3=accountHomepage}(R \times R \times R)$

cross product