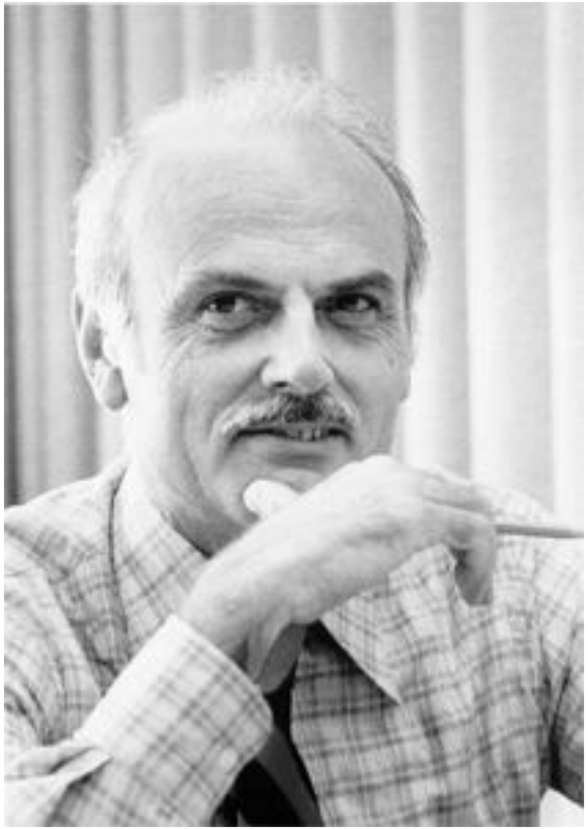
MONTANA
STATE UNIVERSITY

Relational Algebra

Database Theory

Theory

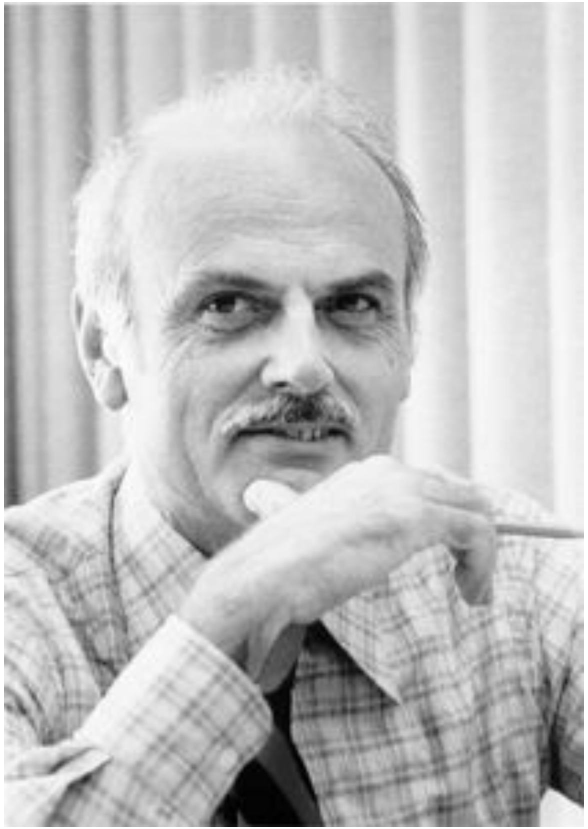
- The relational model grew out of work by E.F. Codd while at IBM
- Codd described a set of *relational algebras* to provide a theoretical basis for SQL



Theory

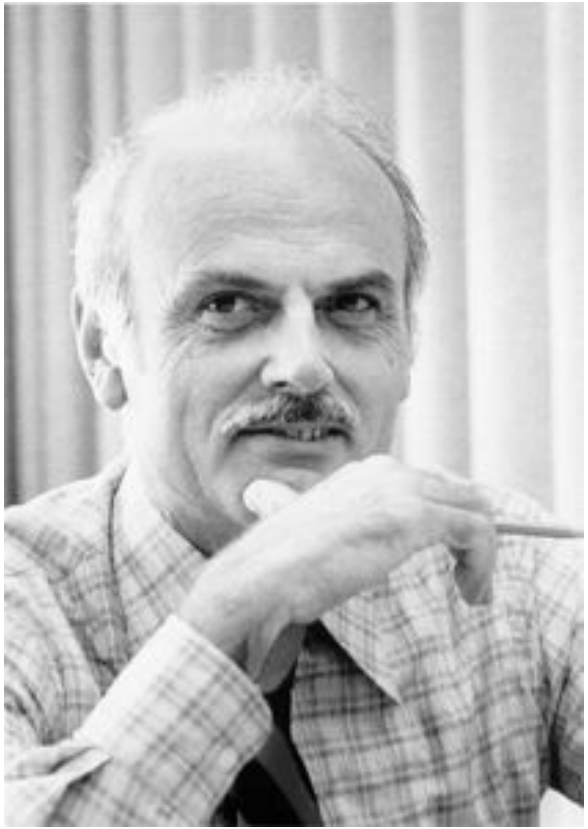
- What is an algebra?

  “In its most general form, algebra is the study of mathematical symbols and the rules for manipulating these symbols”



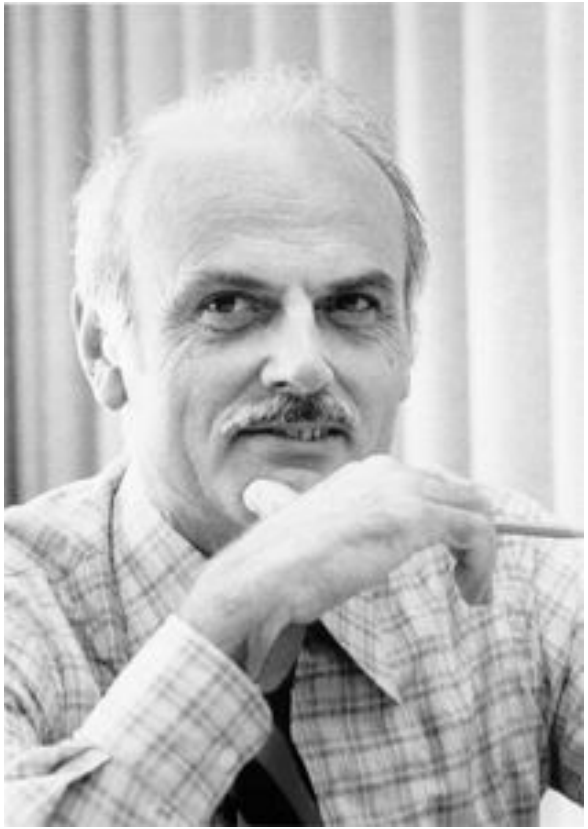
Theory

- Of what use is relational algebra theory?
  - I have to admit, I have never found relational algebra of practical use
  - It apparently can be used at the query level to optimize things, but these always seem obvious to me without a formalization



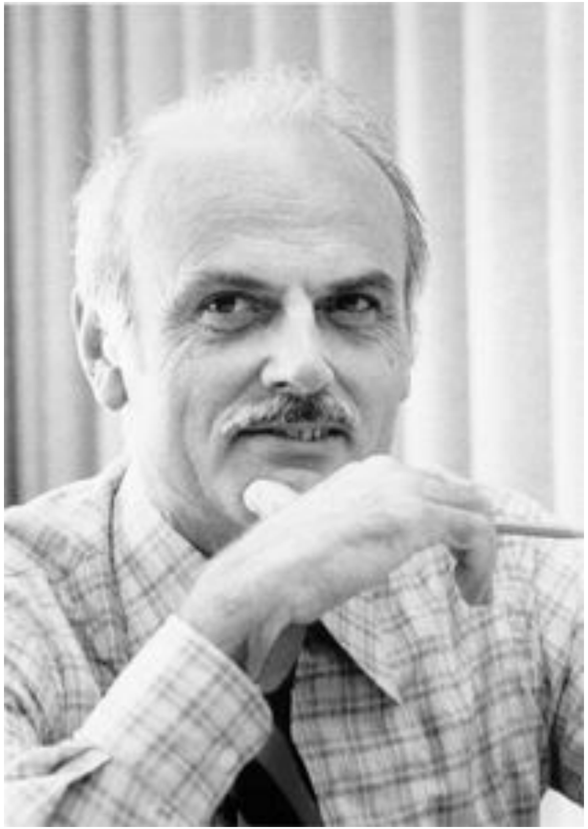
Theory

- OK, so why learn it?
  - It’s a requirement for accreditation
  - If you continue on to 540, you will get much deeper into the theory
    - You can look back and laugh at this lecture...
    - *Never underestimate the motivation of feeling smarter than other people in the tech world*



Theory

- We will focus mainly on the operators of relational algebra, rather than any theory
- Relational algebra works on *relations* rather than *tables*
    - Slight difference between a relation and table:
        - Relations are a *set*
        - Tables are a *bag*

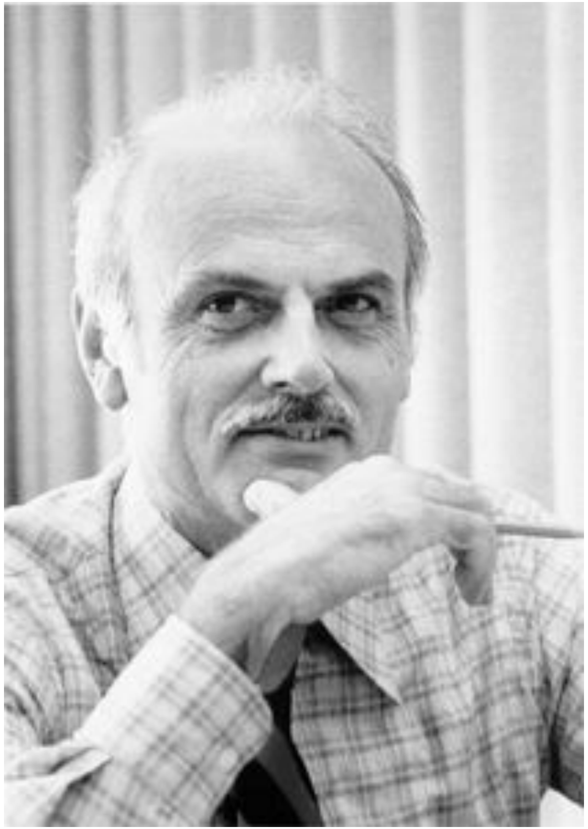


Theory

- What is bag?
  - More properly called a *multiset* or *mset*
  - Allows for multiple instances for each of its elements:

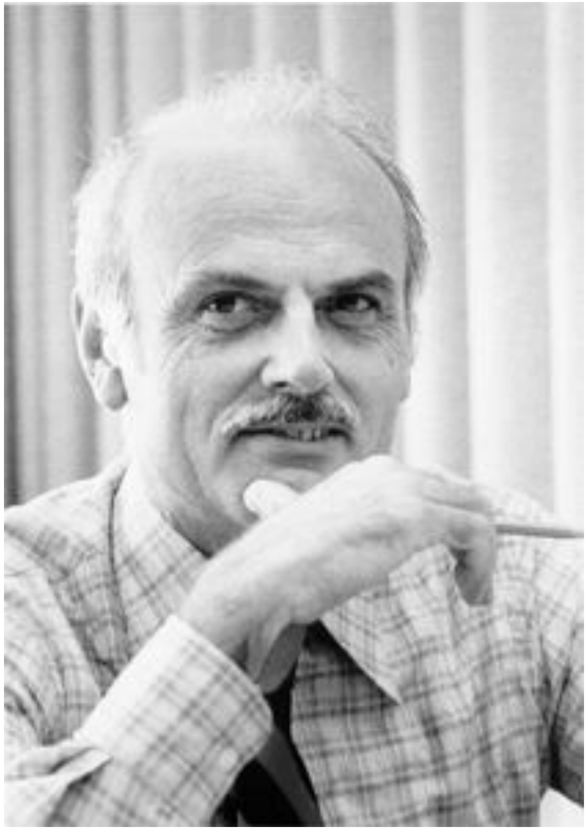
    { A, B, A, C}

  - A great programming problem is bag-equivalence between two lists… *DEMO*



Relational Operators

- Relational Algebra consists of various operators
  - Unary Operations
    - Operations on single relations
  - *Set Operations*
    - Taken from Set Theory, a branch of mathematics
  - Joins and Join-like operators
    - Typically two relations, expressing a *relationship*



Select (σ)

- Used for *selecting* a subset of tuples according to a given *predicate*
- Represented by the sigma(σ) symbol
    - The predicate, annoyingly, is done in subscript

$\sigma_{\text{topic = "Database"}}$ (Tutorials)

$\sigma_{\text{topic = "Database" and author = "guru99"}}$(Tutorials)

$\sigma_{\text{sales > 50000}}$ (Customers)

Project (π)

- eliminates all attributes of the input relation but those specified in the projection list
- Represented by the pi(π) symbol
    - The projection list, again, annoyingly, is done in subscript

$$\Pi_{\texttt{CustomerName, Status}} \texttt{(Customers)}$$

Rename (ρ)

- unary operation used for renaming attributes of a relation
- Represented by the rho(ρ) symbol
    - The projection list, again, annoyingly, is done in subscript

$$\rho_{a/b}(R)$$

Select & Project

- Select & Project correspond to the WHERE clause and the SELECT list, correspondingly

```sql
SELECT name as TrackName
FROM tracks
WHERE Milliseconds > 3 * 60 * 1000;
```

Set Operators - Union

- UNION operator take all tuples from two relations and combines them
- Symbolized by ∪ symbol
- Attribute domains must be compatible
- Duplicates are removed

{A, B} U {A, C} = {A, B, C}

$$R = A \cup B$$

Set Difference

- Set Difference operator returns all symbols in A not in B
- Symbolized by - symbol
- Attribute domains must be compatible

{A, B} - {A, C} = {B}

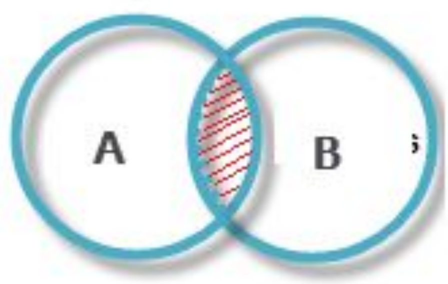
$$R = A - B$$

Set Intersection

- Set Intersection operator returns all symbols in A and in B
- Symbolized by ∩ symbol
- Attribute domains must be compatible

{A, B} ∩ {A, C} = {A}

$$R = A \cap B$$



Cartesian Product

- All possible combinations of rows from relation A and relation B
- Symbolized by X symbol
- Attribute domains need not be compatible with one another
    - Combined with a selection it can be used as a join-like operator

{A, B} X {C, D} = {{A, C}, {A, D},

{B, C}, {B, D}}

$$\sigma_{writer\ =\ 'gauravray'}(Articles\ X\ Notes)$$

Join Operators

- Theta Join
  - General join conditional
  - Again, conditional/predicate is expressed in a subscript

$$A \bowtie_{\theta} B$$

$$A \bowtie_{\texttt{A.column 2 > B.column 2}} (B)$$

Equijoin

- Specialization of the Theta join
- Uses *equality only* for the join condition

$$A \bowtie_{A.\text{column } 2 \,=\, B.\text{column } 2} (B)$$

Natural Join

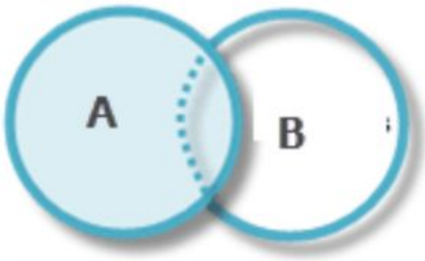
- Omits the conditions
- Implies equijoin on all compatible columns in the relations

$$C \bowtie D$$

Left Outer Join

- Keeps all tuples in left relation
- Only keeps tuples in right relation if there is a match

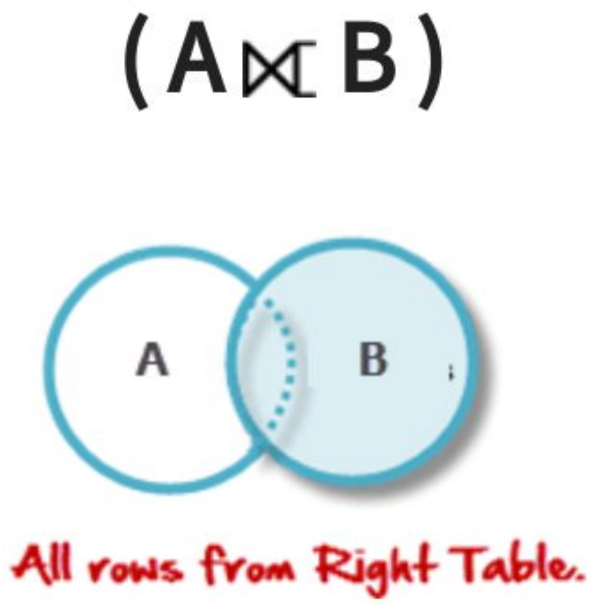
$$(A ⟕ B)$$



All rows from Left Table.

Right Outer Join

- Keeps all tuples in right relation
- Only keeps tuples in left relation if there is a match



Full Outer Join

- Keeps all tuples in both relations
- Even tuples for which there is no match

$$(A ⟗ B)$$

Relational Equations

- Relational equations are collections of relational operators
    - R1 is the member ID of all borrowed books naturally joined with Borrowers with book name Fences
    - R2, same but book name "Inheritance"
    - Final result, the intersection of these two results

$$R1 = \pi_{Member\ ID}(\sigma_{Name="Fences"}(Book \bowtie Borrow))$$

$$R2 = \pi_{Member\ ID}(\sigma_{Name="Inheritance"}(Book \bowtie Borrow))$$

$$R1 \cap R2$$

Relational Calculus

- Another theoretical model to express conditions is with the *Relational Calculus*
- General form is

  { t | COND(t)}

- We are not going to go into the details of this

$$\{\ t \mid \text{EMPLOYEE}(t) \text{ and } t.\text{SALARY} > 50000\ \}$$

OK, so, what's the point?

- I have to be completely frank here: I haven't found a lot of use for the database theory
- Apparently there are some results that come out of both relational calculus and relational algebra that assist with query optimization
- We are briefly covering this there because
    - I want you to have seen the terminology
    - If you go on to 540, these will be a large focus of the class

Relational Algebra

- Today we did a quick tour of relational algebra
  - Looked at the unary operators that correspond closely to the SELECT statement
    - Select (σ)
    - Project (π)
    - Rename (ρ)
- We looked at various set operators
- As well as the JOIN operators
- Finally we took a quick peek at the relational calculus

Quiz Relevance

- On the quiz this week we will have one simple relational algebra expression that you will be asked to describe in english

MONTANA
STATE UNIVERSITY