Relational algebra

What's relational algebra?

- Defines basic operations on *relation instances*
  - composition of operations to form queries
  - basis for SQL
- Useful to represent *execution plans*
  - what are the operations needed to execute a query
  - what is the order of execution of these operations

Basic operations

- Selection σ (choose subset of rows)
- Projection Π (choose subset of columns)
- Cross product $\times$
- Union $\cup$
- Difference $-$
- Rename ρ
- Join $\bowtie$

Set operations

Union, Intersection and Difference

- Same operations as those on any sets
- Apply operations on tuples with same schema

Rename

$$\rho_{R(A1,A2,\ldots)}(S)$$

$$\rho_{Stars(Name,Age,City)}(Actors)$$

Actors (Name, Age, Addr)

Actors

Name	Age	Addr
Priyanka Chopra	42	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Stars

Name	Age	City
Priyanka Chopra	42	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Selection (1/2)

$$R1 = \sigma_C(R2)$$

C is a condition on attributes of *R2*

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return all actors living in Mumbai

$$\sigma_{Addr='Mumbai'}(Actors)$$

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Abhishek Bachchan	42	Mumbai

Selection (2/2)

$$R1 = \sigma_C(R2)$$

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return all actors whose age is more than 36.

$$\sigma_{Age>36}(Actors)$$

Name	Age	Addr
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return all actors whose age is more than 36 and who live in Mumbai

$$\sigma_{Age>36\ and\ Addr='Mumbai'}(Actors)$$

Name	Age	Addr
Abhishek Bachchan	42	Mumbai

Projection (1/2)

$$R1 = \Pi_L(R2)$$

Actors

Name	**Age**	**Addr**
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return the name and age of all actors

$$\Pi_{Name,Age}(Actors)$$

Name	**Age**
Priyanka Chopra	36
Anthony Hopkins	81
Bill Nighy	69
Abhishek Bachchan	42

Projection (2/2)

$$R1 = \Pi_L(R2)$$

Actors

Name	**Age**	**Addr**
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return the addresses of the actors

$$\Pi_{Addr}(Actors)$$

Addr
Mumbai
LA

Duplicate elimination under set semantics

Cross product

$$R3 = R1 \times R2$$

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Movies

Name	Year	Title
Priyanka Chopra	2011	Don-II
Anthony Hopkins	2011	MI-IV
Bill Nighy	2009	Valkyrie
Abhishek Bachchan	2010	Raavan

Actor.name	Age	Addr	Movies.Name	Year	Title
Priyanka Chopra	36	Mumbai	Priyanka Chopra	2011	Don-II
…15 more rows…					

Joins (1/2)

$$R3 = R1 \bowtie_C R2$$

C is a condition on attributes of *R1* and/or *R2*

Equi-join

Actors.name = Movies.name

Actors

Name	Age	Addr
PC	36	Mumbai
AH	81	LA
BN	69	LA
AB	42	Mumbai

Movies

Name	Year	Title
PC	2011	Don-II
AH	2011	Thor: R
BN	2009	Valkyrie
AB	2010	Raavan

Return all information about actors and their movies

$$Actors \bowtie_{A.Name=M.Name} Movies$$

Name	Age	Addr	Year	Title
PC	36	Mumbai	2011	Don-II
AH	81	LA	2011	Thor: R
BN	69	LA	2009	Valkyrie
AB	42	Mumbai	2010	Raavan

Joins (2/2)

$R_1 \bowtie R_2$

- Natural joins
  - implicitly compares attributes of the same name for equality
- Theta join
  - conditions not restricted to equality
- Left-outer/right-outer/full-outer joins
  - non-matching tuples are still returned
- Self-join
  - table joining with itself

Left outer joins

$$R = X ⟕ Y$$

Actors

Name	Age	Addr
PC	36	Mumbai
AH	81	LA
BN	69	LA
AB	42	Mumbai

Movies

Name	Year	Title
PC	2011	Don-II
AH	2011	Thor: R
AB	2010	Raavan

Return all information about actors and their movies

$$Actors ⟕_{A.Name=M.Name} Movies$$

Name	Age	Addr	Year	Title
PC	36	Mumbai	2011	Don-II
AH	81	LA	2011	Thor: R
BN	69	LA	*null*	*null*
AB	42	Mumbai	2010	Raavan

Actors ⋈ Actors_1

Actors.parent = Actors_1.name | Nar GC, GP

Actors.name = Actors_1.parent | GP, GC

Self Join

Return all grandparents and their grand children

Actors

Name	Age	Addr	Parent
PC	36	Mumbai	Madhu
AH	81	LA	Muriel
BN	69	LA	Catherine
AB	42	Mumbai	Jaya
Jaya	63	Mumbai	Indira

Actors_1

Name	Age	Addr	Parent
PC	36	Mumbai	Madhu
AH	81	LA	Muriel
BN	69	LA	Catherine
AB	42	Mumbai	Jaya
Jaya	63	Mumbai	Indira

AB Indira

Composition of operators (1/2)

$\sigma_{Age>36}(\Pi_{N,Addr}(Actors))$

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Return the names and addresses of actors over 35

R1

$$\Pi_{Name,Addr}(\sigma_{Age>36}(Actors))$$

Return the names of actors over 35 who live in Mumbai

$$\Pi_{Name}(\sigma_{Age>36\ and\ Addr='Mumbai'}(Actors))$$

Composition of operators (2/2)

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Movies

Name	Year	Title
Priyanka Chopra	2011	Don-II
Anthony Hopkins	2011	MI-IV
Bill Nighy	2009	Valkyrie
Abhishek Bachchan	2010	Raavan

Return the names of actors below the age of 50 who have acted in a movie in 2011

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\begin{aligned} Allmovies &= Actors \bowtie_{A.Name=M.Name} Movies \\ Movies1 &= \sigma_{Age<50 \text{ AND } Year=2011}(AllMovies) \\ Result &= \Pi_{Name}(Movies1) \end{aligned}$$

$3 \times (4+5)$

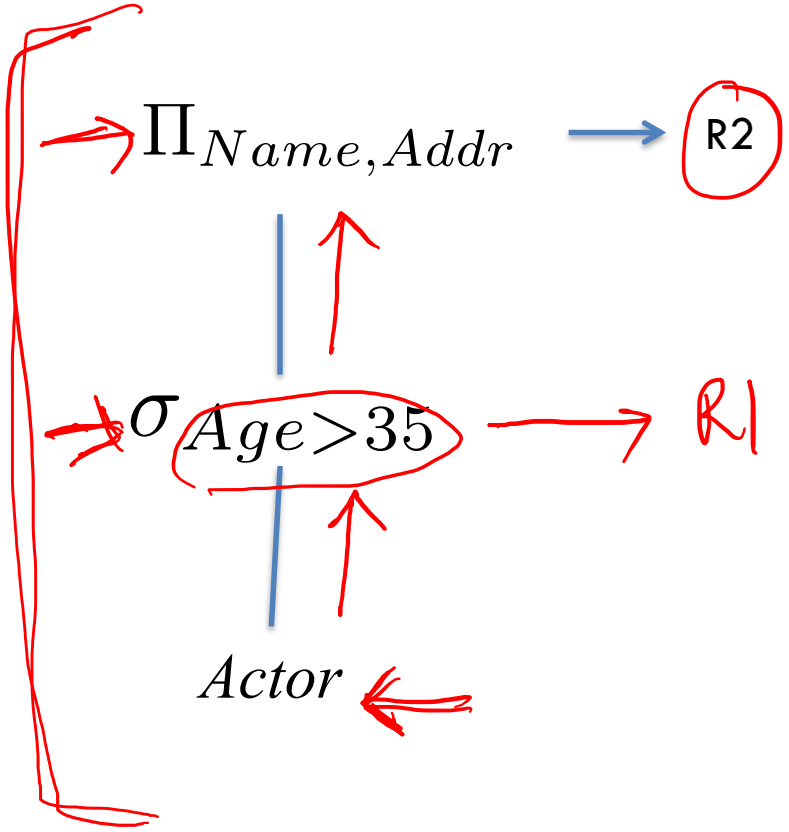
Notation (1/2)

Return the names and addresses of actors over 35

$$\Pi_{Name,Addr}(\sigma_{Age>35}(Actor))$$

$$R1 = \sigma_{Age>35}(Actor)$$

$$R2 = \Pi_{Name,Addr}(R1)$$

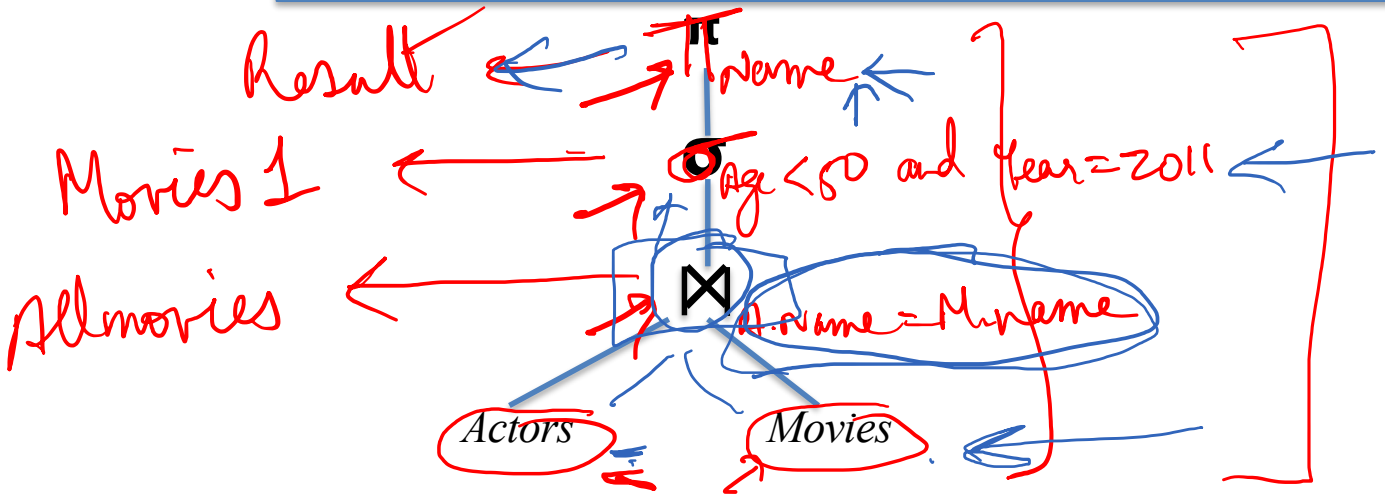


Notation (2/2)

Return the names of actors below the age of 50 who have acted in a movie in 2011

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\begin{aligned} Allmovies &= Actors \bowtie_{A.Name=M.Name} Movies \\ Movies1 &= \sigma_{Age<50 \text{ AND } Year=2011}(AllMovies) \\ Result &= \Pi_{Name}(Movies1) \end{aligned}$$



Relational algebra for bags

- Efficiency issues if we consider relations as sets
  - Extra effort to eliminate duplicates
- Select, project, join (SPJ) work exactly the same
  - Applied to one tuple at a time
- Set operations become bag operations
  - Need to be careful about semantics
  - Union, Intersection, Difference

More operators

- Duplicate elimination
- Extended projection
- Aggregation
  - count, min, max, sum, avg
- Grouping
- Sorting

$\boxed{d(R)}$

delta

g Gamma

t Tau

Extended Projection

- Same as projection, but some operations can be "directly" expressed

Prices

City	Petrol	Tax
Bangalore	39.2	33.2
Delhi	38.2	31.5
Mumbai	41.47	30

Return the total price of petrol

$$\Pi_{City, Petrol+Tax}(Prices)$$

City	Petrol+Tax
Bangalore	72.4
Delhi	69.7
Mumbai	71.47

Aggregation and grouping (1/2)

- Grouping $\boxed{\gamma_L(R)}$ unary
  - L is a list of grouping attributes and/or aggregate operators

Movies

Movie	City	Boxoffice
MI-IV	LA	2,000,000
Don-II	LA	500,000
MI-IV	NY	3,000,000

Return total boxoffice returns per movie

aggregate

aggregation

grouping attribute

$$\gamma_{Movie,Sum(Boxoffice)}(Movies)$$

grouping attribute

Movie	City	Boxoffice
MI-IV	LA	2,000,000
MI-IV	NY	3,000,000
Don-II	LA	500,000

Movie	Boxoffice
MI-IV	5,000,000
Don-II	500,000

Aggregation and grouping (2/2)

- Grouping $\gamma_L(R)$
  - L is a list of grouping attributes and/or aggregate operators

Movie	City	Boxoffice
MI-IV	LA	2,000,000
Don-II	LA	500,000
MI-IV	NY	3,000,000

Return total boxoffice returns per city

aggregate

grouping attribute

$$g_{City,Sum(Boxoffice)}(Movies)$$

Movie	City	Boxoffice
MI-IV	LA	2,000,000
Don-II	LA	500,000
MI-IV	NY	3,000,000

City	Boxoffice
LA	2,500,000
NY	3,000,000

(Movies)

Sum(Boxoffice)

Boxoffice

5,500,000