Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- Union: $R \cup S$
- Difference: $R - S$

Derived Operators

- Intersection: $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- Natural join: $R \bowtie S$
- Renaming: $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

- New Problem:
  - The User table and the Member table share one column (UID)
  - Can we simplify the join predicate, if two tables join on all shared columns?

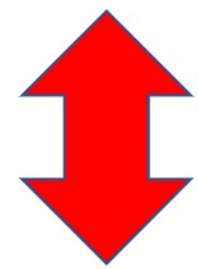
User

uid	name	age	p
142	Bart	10	0
123	Milhouse	10	0
857	Lisa	8	0
456	Ralph	8	0
…	…	…	…

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

$User \bowtie_{User.uid = Member.uid} Member$



$User \bowtie Member$

Derived operator: Natural Join

- Input: two tables R and S
- Notation: $R \bowtie S$
- Purpose: relate rows from two tables
  - Enforce equality between identically named columns
  - Eliminate one copy of identically named columns
- Shorthand for $\pi_L(R \bowtie_p S)$, where
  - p equates each pair of columns common to R and S
  - L is the union of column names from R and S (with duplicate columns)

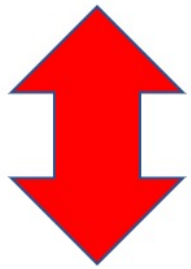
User

uid	name	age	p
142	Bart	10	0
123	Milhouse	10	0
857	Lisa	8	0
456	Ralph	8	0
…	…	…	…

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

$User \bowtie_{User.uid = Member.uid} Member$



$User \bowtie Member$

Corresponding SQL:

Select * FROM User, Member **WHERE** User.uid = Member.uid

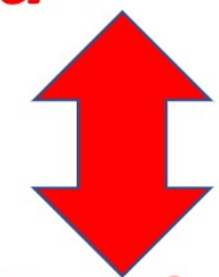
Select * FROM User JOIN Member ON User.uid = Member.uid

Select * FROM User NATURAL JOIN Member

Corresponding SQL 1:

Select * **FROM** User, Member **WHERE** User.uid = Member.uid

$$\sigma_{User.uid = Member.uid}(User \times Member)$$



$$User \bowtie_{User.uid = Member.uid} Member$$

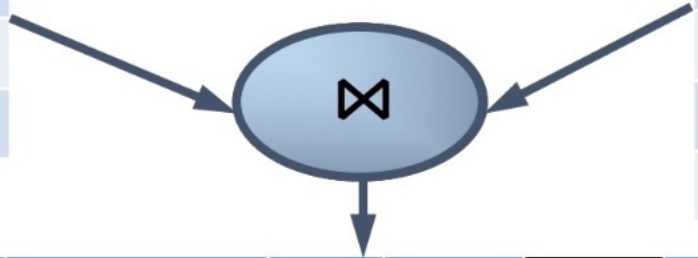
Corresponding SQL 2:

Select * **FROM** User JOIN Member ON User.uid = Member.uid

Natural Join Example

$$User \bowtie Member = \pi_{?}(User \bowtie_{?} Member)$$
$$= \pi_{uid,name,age,pop,gid}\left(User \bowtie_{\substack{User.uid= \\ Member.uid}} Member\right)$$

uid	name	age	pop
123	Milhouse	10	0.2
857	Lisa	8	0.7
…	…	…	…



uid	gid
123	gov
857	abc
857	gov
…	…

uid	name	age	pop	[redacted]	gid
123	Milhouse	10	0.2	[redacted]	gov
[redacted]					
[redacted]					
[redacted]					
857	Lisa	8	0.7	[redacted]	abc
857	Lisa	8	0.7	[redacted]	gov
…	…	…	…	[redacted]	…

Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- Union: $R \cup S$
- Difference: $R - S$

Derived Operators

- Intersection: $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- ~~Natural join:~~ $R \bowtie S$
- Renaming: $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

Union Example

- Q: find all students (part or full time)
- A: PT-STUDENT union FT-STUDENT

FT-STUDENT

Ssn	Name	Address
129	peters	main str
239	lee	5th ave

PT-STUDENT

Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

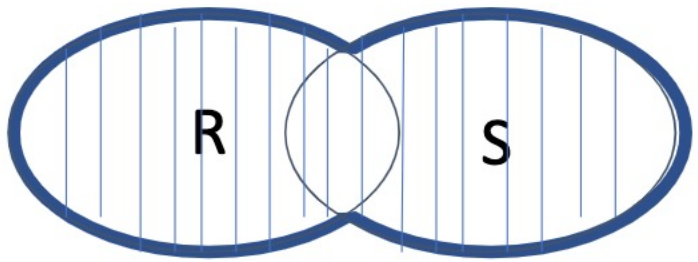
U

SSN	Name	Address
129	Peters	Main str
239	Lee	5th ave
123	Smith	Main Str
234	Jones	Forbes Ave

Two tables are ‘union compatible’ if they have the same attributes (‘domains’)

Union

- Input: two tables R and S
- Notation: R ∪ S
  - R and S must have identical schema
- Output:
  - Has the same schema as R and S
  - Contains all rows in R and all rows in S (with duplicate rows removed)



Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- ~~Union:~~ $R \cup S$
- Difference: $R - S$

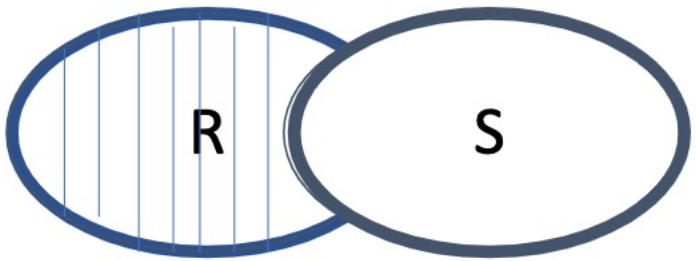
Derived Operators

- Intersection: $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- ~~Natural join:~~ $R \bowtie S$
- Renaming: $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

Difference

- Input: two tables R and S
- Notation: R – S
  - R and S must have identical schema
- Output:
  - Has the same schema as R and S
  - Contains all rows in R that are not in S



Difference Example

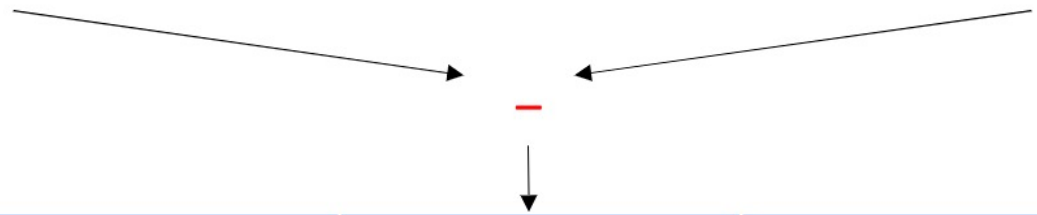
- Q: find all students who is not a staff
- A: Student - Staff

Student

SSNc	Name	Address
129	Peters	Main str
239	Lee	5th ave
123	Smith	Main Str
234	Jones	Forbes Ave

Staff

SSN	Name	Address
129	Peters	Main str
239	Lee	5th ave
323	Sarah	Sunset Blvd
432	Mary	Sunset Blvd



SSNc	Name	Address
123	Smith	Main Str
234	Jones	Forbes Ave

Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- ~~Union:~~ $R \cup S$
- ~~Difference:~~ $R - S$

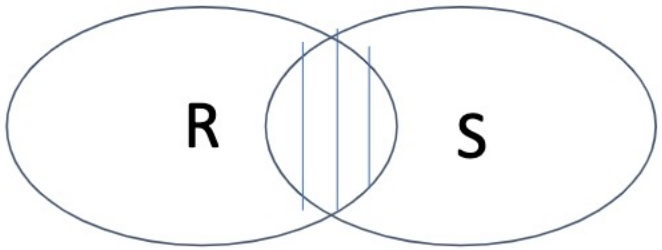
Derived Operators

- Intersection: $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- ~~Natural join:~~ $R \bowtie S$
- Renaming: $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

Derived Operator: Intersection

- Input: two tables R and S
- Notation: R ⋂ S
  - R and S must have identical schemas
- Output:
  - Has the same schema as R and S
  - Contains all rows that are in both relations
- Shorthand for R – (R – S)
- Also equivalent to S – (S – R)
- And to R ⋈ S



Intersection Example

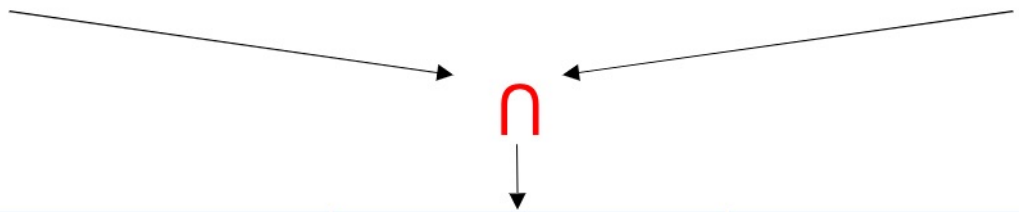
- Student intersection Staff
- Student – (Student – Staff)

Student

SSN	Name	Address
129	Peters	Main str
239	Lee	5th ave
123	Smith	Main Str
234	Jones	Forbes Ave

Staff

SSN	Name	Address
129	Peters	Main str
239	Lee	5th ave
323	Sarah	Sunset Blvd
432	Mary	Sunset Blvd



SSN	Name	Address
129	Peters	Main Str
239	Lee	5th ave

Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- ~~Union:~~ $R \cup S$
- ~~Difference:~~ $R - S$

Derived Operators

- ~~Intersection:~~ $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- ~~Natural join:~~ $R \bowtie S$
- Renaming: $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

Renaming

- Input: a table R
- Notation: ρ_s R, $\rho_{(A1, A2, \ldots)}$R, or $\rho_{S(A1, A2, \ldots)}$R
- Purpose: "rename" a table or its columns
- Output: a table with the same rows as R, but called differently
- Used to
  - Avoid confusion caused by identical column names
  - Create identical column names for natural joins
- It doesn't modify the database
  - Think of the renamed table as a copy of the original

Renaming Example

- IDs of users who belong to at least two groups

$$Member \bowtie_{?} Member$$

$$\pi_{uid}\left(Member \bowtie_{\substack{Member.uid=Member.uid \wedge \\ Member.gid \neq Member.gid}} Member\right)$$

WRONG!

$$\pi_{uid_1}\left(\begin{array}{c}\rho_{(uid_1,gid_1)}Member \\ \bowtie_{uid_1=uid_2 \wedge gid_1 \neq gid_2} \\ \rho_{(uid_2,gid_2)}Member\end{array}\right)$$

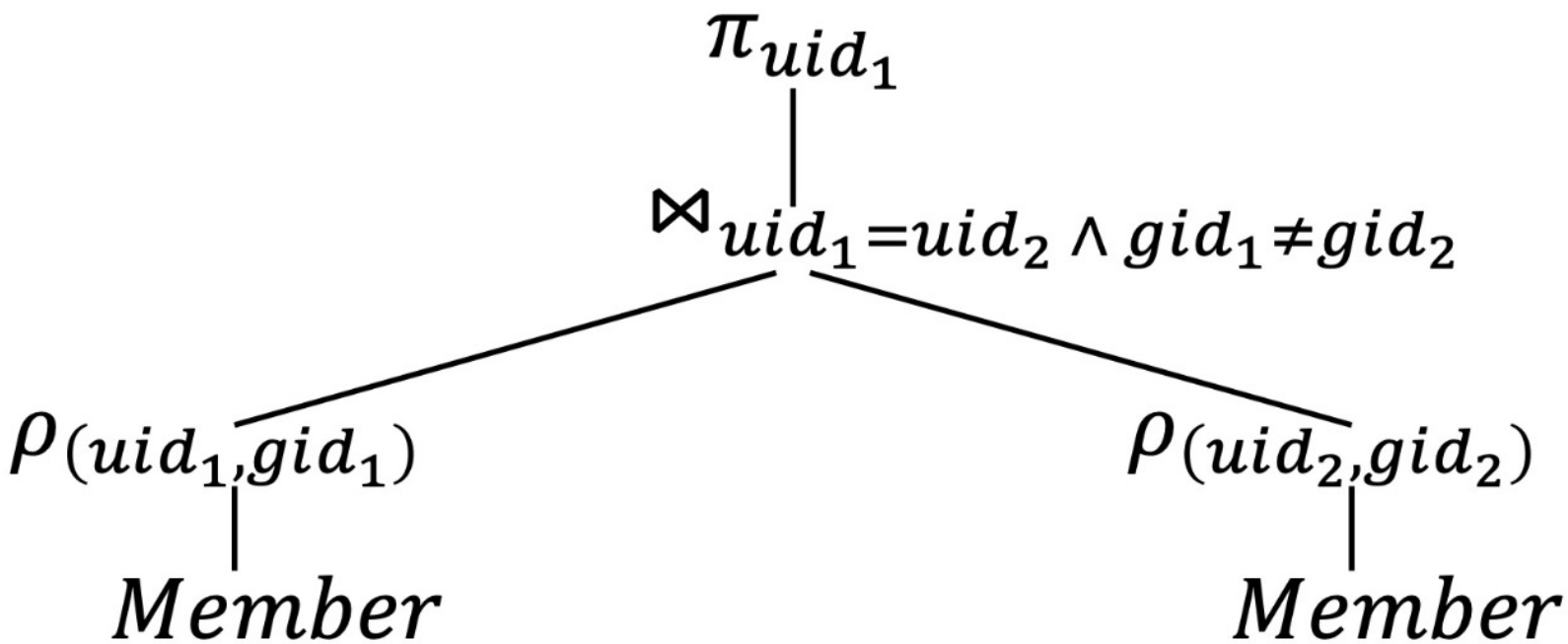
Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

Expression Tree Notation

Member

uid	*gid*
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…



Relational Operators

Fundamental Operators

- ~~Selection:~~ $\sigma_p R$
- ~~Projection:~~ $\pi_L R$
- ~~Cross product:~~ $R \times S$
- ~~Union:~~ $R \cup S$
- ~~Difference:~~ $R - S$

Derived Operators

- ~~Intersection:~~ $R \cap S$
- ~~Join:~~ $R \bowtie_p S$
- ~~Natural join:~~ $R \bowtie S$
- ~~Renaming:~~ $\rho_{S(A_1,A_2,\dots)} R$

Compose operators to make complex queries.

An exercise: Names of users in Lisa's groups

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

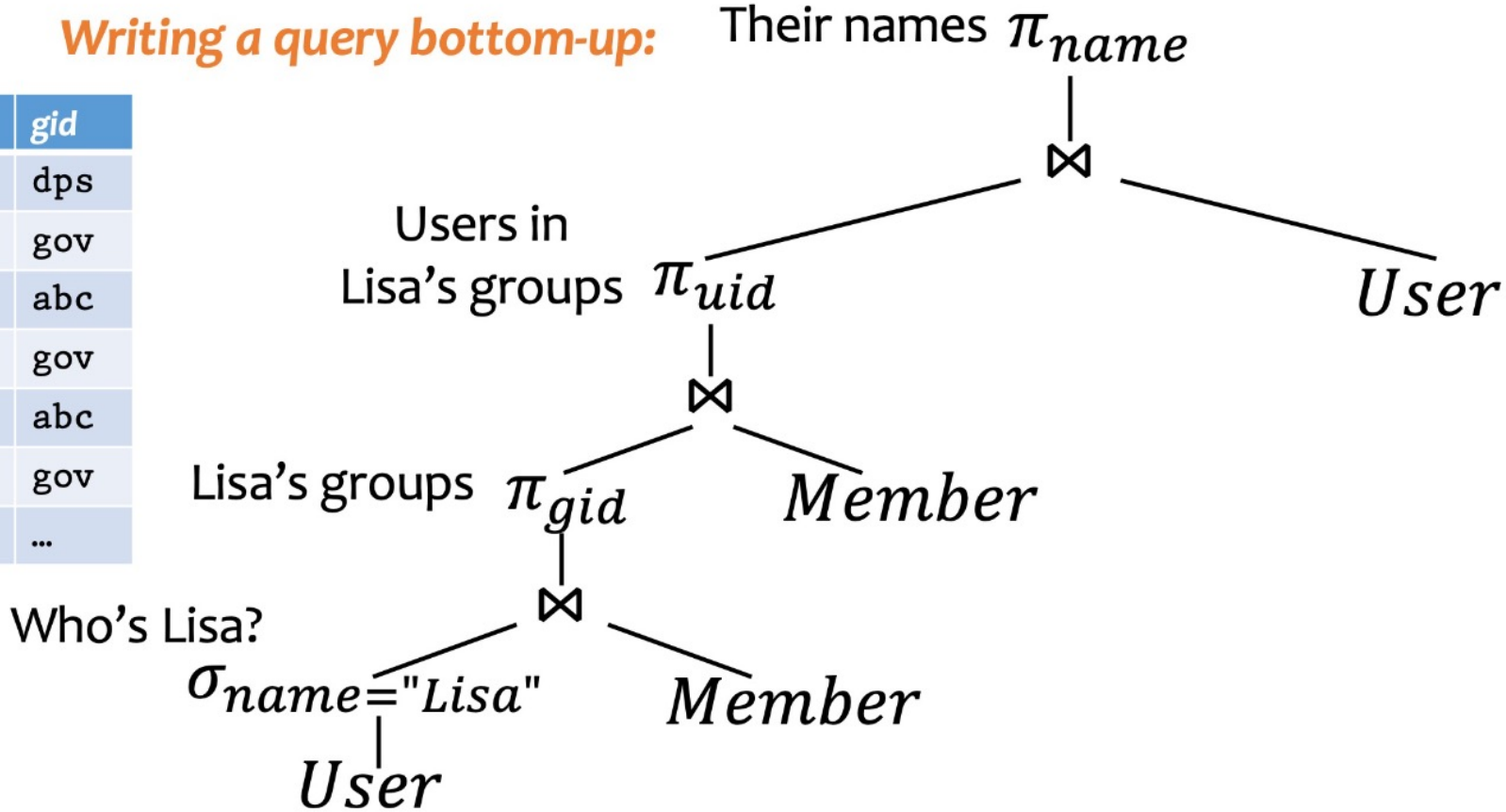
Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

Member

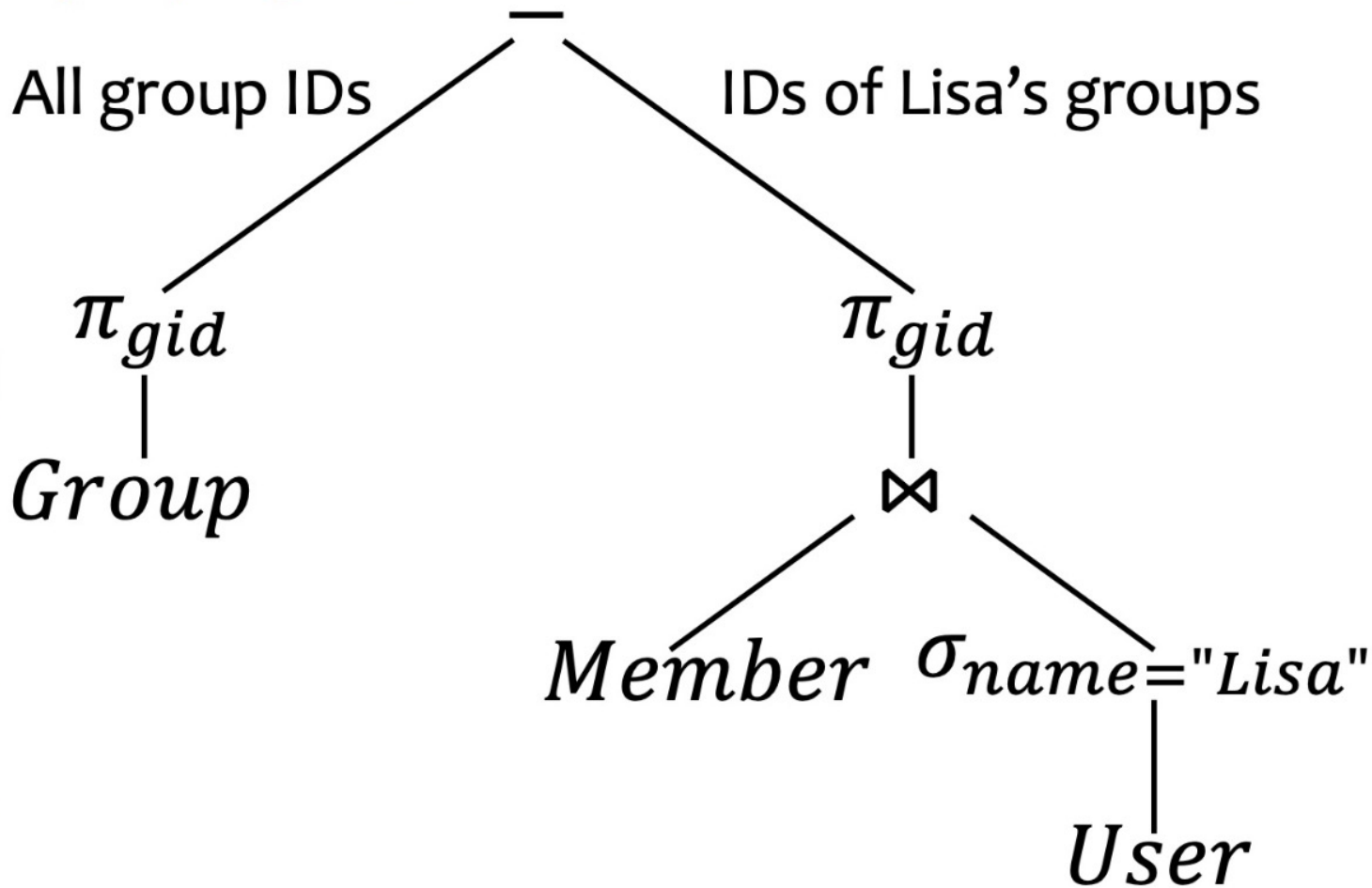
uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

Writing a query bottom-up:



Another exercise: IDs of groups that Lisa doesn't belong to

Writing a query top-down:



User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

More Examples

STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

CLASS		
c-id	c-name	units
413	s.e.	2
412	o.s.	2

TAKES		
SSN	c-id	grade
123	413	A
234	413	B

- Find course names of ‘smith’
- Find ssn of ‘overworked’ students, ie., who take 412, 413, 415

Find course names of ‘Smith’

$$\pi_{c-name}[\,\sigma_{name='smith'}(\,STUDENT \bowtie TAKES \bowtie CLASS\,)]$$

Find ssn of ‘overworked’ students, ie., who take 412, 413, 415

$$\sigma_{c-name=412}(TAKES) \cap$$
$$\sigma_{c-name=413}(TAKES) \cap$$
$$\sigma_{c-name=415}(TAKES)$$