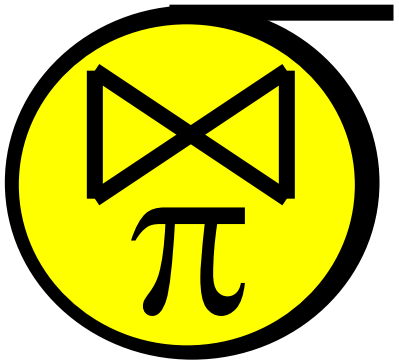
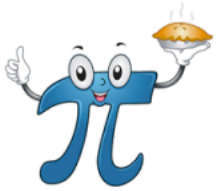
Information Storage and Management I

Dr. Alejandro Arbelaez



Relational Algebra

Relational Algebra



- Basic operations:
  - ***Selection*** (σ) Selects a subset of rows from relation.
  - ***Projection*** (π) Deletes unwanted columns from relation.
  - ***Cross-product*** ($\times$) Allows us to combine two relations.
  - ***Set-difference*** (-) Tuples in reln. 1, but not in reln. 2.
  - ***Union*** ($\cup$) Tuples in reln. 1 and in reln. 2.
  - ***Renaming*** (ρ) (for named perspective)
- Additional operations:
  - Intersection, ***join***, division, renaming: Not essential, but (very!) useful.
- Since each operation returns a relation, operations can be *composed*!

Joins

- ***Condition Join***: $R \bowtie_C S = \sigma_C (R \times S)$

	(sid)	sname	rating	age	(sid)	bid	day
X	22	dustin	7	45.0	22	101	10/10/96
	22	dustin	7	45.0	58	103	11/12/96
X	31	lubber	8	55.5	22	101	10/10/96
	31	lubber	8	55.5	58	103	11/12/96
X	58	rusty	10	35.0	22	101	10/10/96
X	58	rusty	10	35.0	58	103	11/12/96

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	58	103	11/12/96

$$S1 \bowtie_{S1.sid < R1.sid} R1$$

- ***Result schema*** same as that of cross-product.
- Fewer tuples than cross-product, might be able to compute more efficiently
- Sometimes called a ***theta-join***.

Joins

- ***Equi-Join***: A special case of condition join where the condition *c* contains only ***equalities*** and ^.

sid	sname	rating	age	bid	day
22	dustin	7	45.0	101	10/10/96
58	rusty	10	35.0	103	11/12/96

$$S1 \bowtie_{sid} R1$$

- ***Result schema*** similar to cross-product, but only one copy of fields for which equality is specified.
- ***Natural Join***: Equijoin on *all* common fields.

Lab Solution

Construct the Natural Join of the following two tables

A1	A2
1	Red
2	Blue
3	Black

A1	B2	B3
1	BMW	1972
2	Honda	1968
4	Ferrari	1978

$$Table1 \bowtie Table2$$

A1	A2	B2	B3
1	Red	BMW	1972
2	Blue	Honda	1968

Lab Solution

Translate the following join expression in SQL to relational Algebra (show parse tree of the expression)

SELECT TableA.A1, TableA.A2, TableA.A3, TableB.B2 **FROM** TableA, TableB **WHERE**
TableA.A2 > TableB.B1 AND TableB.B2 = TableA.A2

$$\pi_{A1,A2,A3}(TableA \bowtie_{A2>B1 \wedge B2=A2} TableB)$$

Lab Solution

Consider the following relational DB

employee(person_name, street, city)

works(person_name, company_name, salary)

company(company_name, city)

manager(person_name, manager_name)

A. Find the names of all employees in this database who work for “First Bank Corporation”

$$\pi_{person\_name}\left(\sigma_{company\_name=\text{"}FirstBankCorporation\text{"}}\left(works\right)\right)$$

B. Find the names and cities of residence of all employees who work for First Bank Corporation

$$\pi_{person\_name,city}(employee \bowtie \sigma_{company\_name=\text{"}FirstBankCorporation\text{"}}(works))$$

C. Find the names of all employees who live in the same city and on the same street as do their managers

$$\rho(emp2, employee)$$

$$\rho(temp1, (manager \times employee \times emp2))$$

$$\rho(temp2, \sigma_{manager.person\_name=employee.person\_name \wedge manager.manager\_name=emp2.person\_name}(temp1))$$

$$\rho(temp3, \sigma_{employee.stree=emp2.stree \wedge employee.city=emp2.city}(temp2))$$

Find names of sailors who've reserved boat #103

Solution 1:

$$\pi_{sname}((\sigma_{bid=103} Reserves) \bowtie Sailors)$$

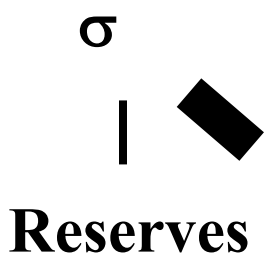
Solution 3:

$$\pi_{sname}(\sigma_{bid=103}(Reserves \bowtie Sailors))$$

Which one to choose?
and why?

Find names of sailors who've reserved boat #103

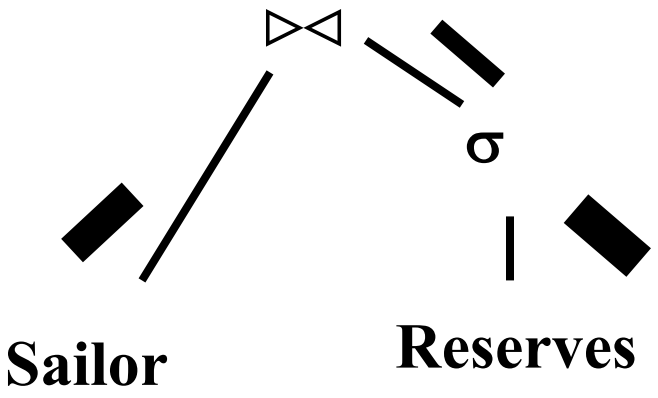
$$\pi_{sname}((\sigma_{bid=103} Reserves) \bowtie Sailors)$$



Query Tree Plan

Find names of sailors who've reserved boat #103

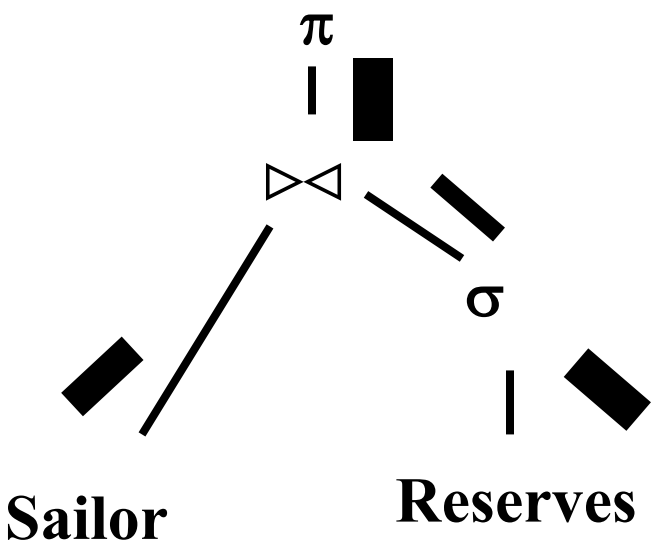
$$\pi_{sname}((\sigma_{bid=103} Reserves) \bowtie Sailors)$$



Query Tree Plan

Find names of sailors who've reserved boat #103

$$\pi_{sname}((\sigma_{bid=103} Reserves) \bowtie Sailors)$$

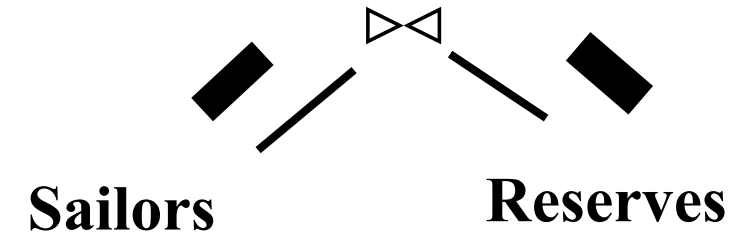


Query Tree Plan

Find names of sailors who've reserved boat #103

Query Tree Plan

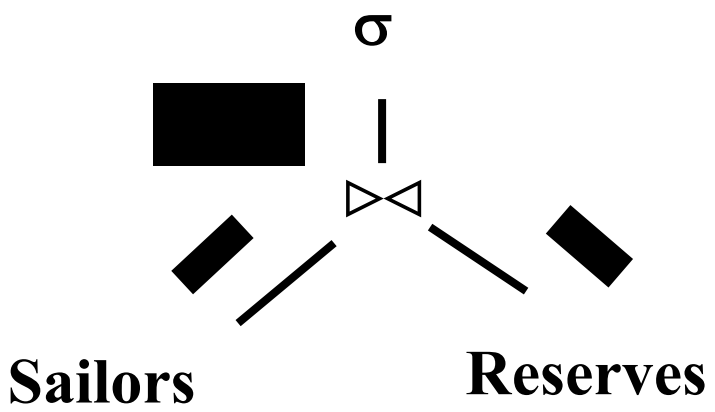
$$\pi_{sname}(\sigma_{bid=103}(Reserves \bowtie Sailors))$$



Find names of sailors who've reserved boat #103

Query Tree Plan

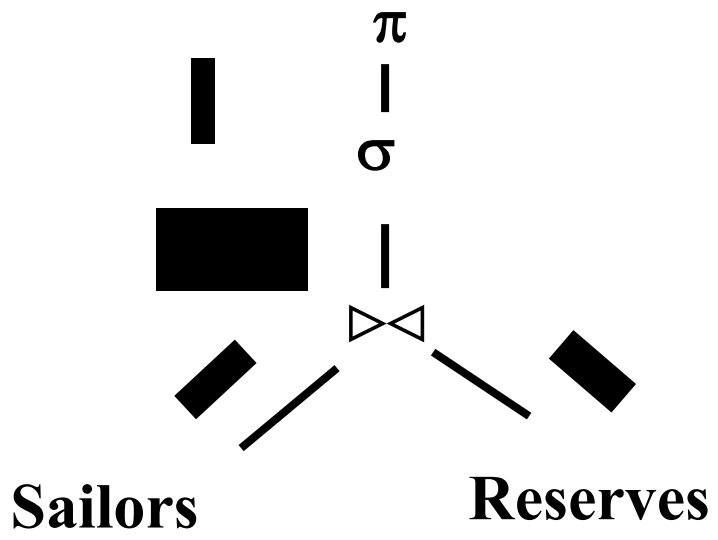
$$\pi_{sname}(\sigma_{bid=103}(Reserves \bowtie Sailors))$$



Find names of sailors who've reserved boat #103

Query Tree Plan

$$\pi_{sname}(\sigma_{bid=103}(Reserves \bowtie Sailors))$$

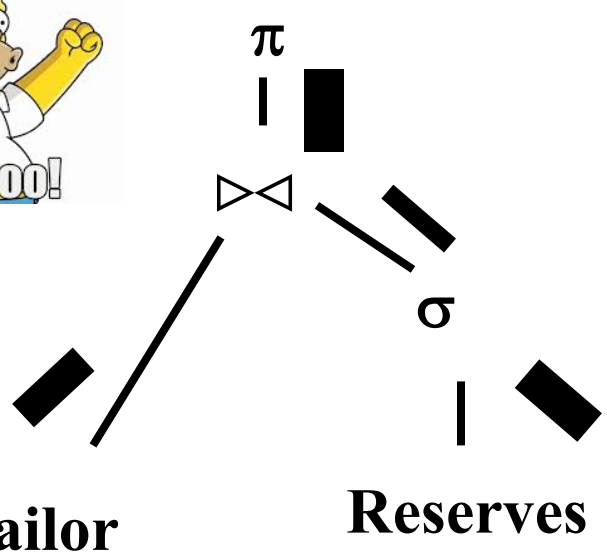


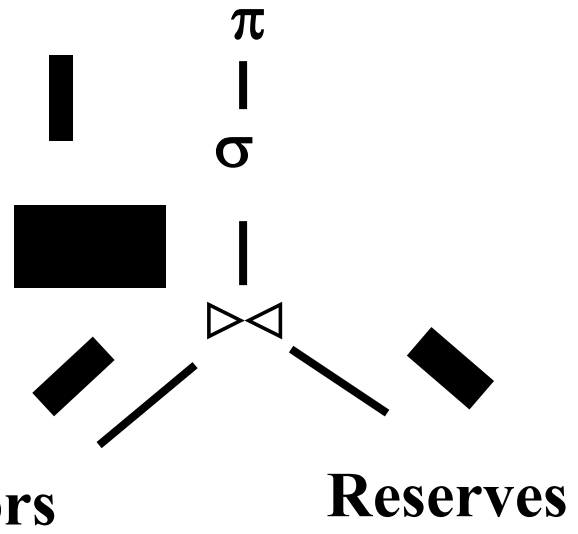
Find names of sailors who've reserved boat #103

OBSERVATION: try to perform selections and projections early!

$$\pi_{sname}((\sigma_{bid=103}\text{Reserves}) \bowtie Sailors)$$

$$\pi_{sname}(\sigma_{bid=103}(\text{Reserves} \bowtie Sailors))$$





Find names of sailors who've reserved a red boat

$$\pi_{sname}((\sigma_{color='red'} Boats) \bowtie Reserves \bowtie Sailors)$$

Sid	Sname	Rating	Age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

Information about boat color only available in Boats; so need an extra join:

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

Information about boat color only available in Boats; so need an extra join:

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

$$\sigma_{color='red'} Boats$$

Find names of sailors who've reserved a red boat

Information about boat color only available in Boats; so need an extra join:

$$(\sigma_{color='red'} Boats) \bowtie \text{Reserves}$$

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

$$((\sigma_{color='red'} Boats) \bowtie Reserves \bowtie Sailors)$$

Sid	Sname	Rating	Age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

$$\pi_{sname}((\sigma_{color='red'} Boats) \bowtie Reserves \bowtie Sailors)$$

Sid	Sname	Rating	Age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

Information about boat color only available in Boats; so need an extra join:

$$\pi_{sname}((\sigma_{color='red'} Boats) \bowtie \text{Re}serves \bowtie Sailors)$$

Alternative solution:

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'} Boats) \bowtie \text{Re}s) \bowtie Sailors)$$

Find names of sailors who've reserved a red boat

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'}Boats) \bowtie Res) \bowtie Sailors)$$

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Find names of sailors who've reserved a red boat

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'}Boats) \bowtie Res) \bowtie Sailors)$$

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Find names of sailors who've reserved a red boat

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'} Boats) \bowtie \text{Res}) \bowtie Sailors)$$

Bid	Bname	Color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

An Instance B1 of Boats

Sid	Bid	Day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98
31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

An Instance R2 of Reserves

Sid	Sname	Rating	Age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Find names of sailors who've reserved a red boat

Information about boat color only available in Boats; so need an extra join:

$$\pi_{sname}((\sigma_{color='red'} Boats) \bowtie \text{Re}serves \bowtie Sailors)$$

A more efficient solution:

$$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color='red'} Boats) \bowtie \text{Re}s) \bowtie Sailors)$$

A query optimizer can find this given the first solution!

Find sailors who've reserved a red or a green boat

Can identify all red or green boats, then find sailors who've reserved one of these boats:

$$\rho\ (Tempboats, (\sigma_{color='red' \vee color='green'}\ Boats))$$

$$\pi_{sname}(Tempboats \bowtie \text{Re}serves \bowtie Sailors)$$

Can also define Tempboats using union! (How?)

What happens if $\vee$ is replaced by $\wedge$ in this query?

Find sailors who've reserved a red **and** a green boat

Previous approach won't work! Must identify sailors who've reserved red boats, sailors who've reserved green boats, then find the intersection (note that *sid* is a key for Sailors):

Find sailors who've reserved a red **and** a green boat

Previous approach won't work! Must identify sailors who've reserved red boats, sailors who've reserved green boats, then find the intersection (note that *sid* is a key for Sailors):

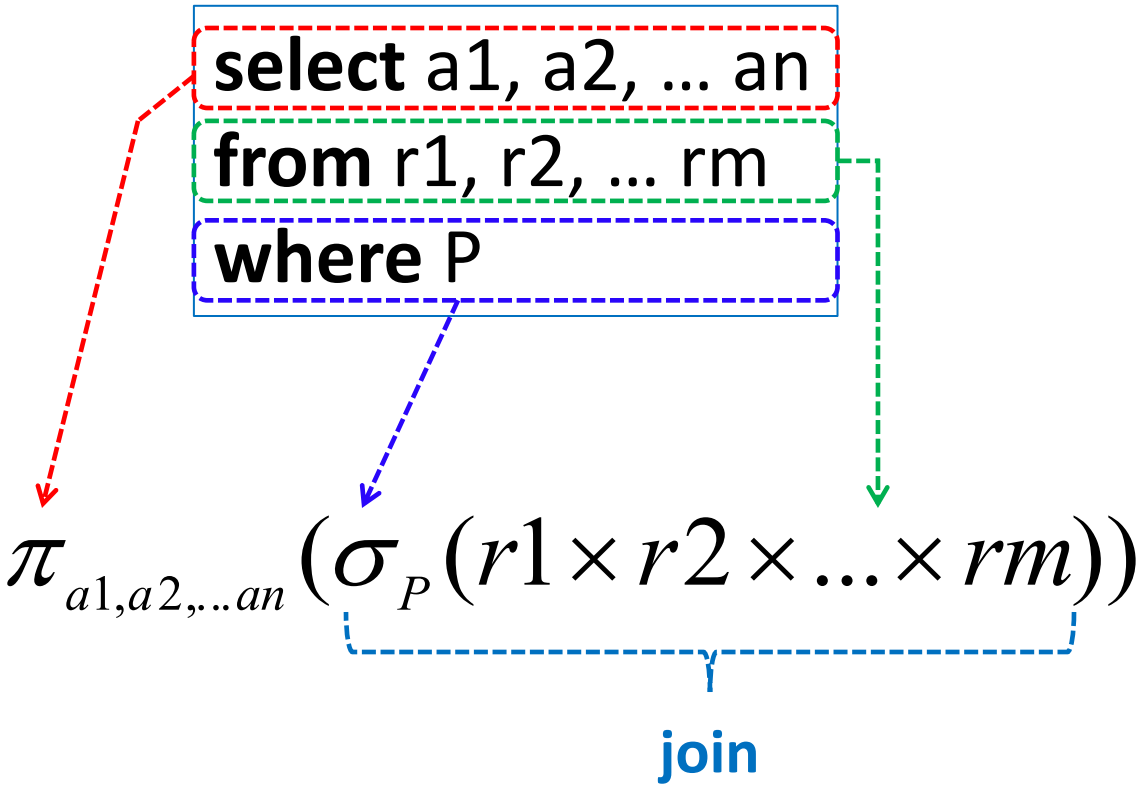
$$\rho\ (Tempred, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie \text{Re}serves))$$

$$\rho\ (Tempgreen, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie \text{Re}serves))$$

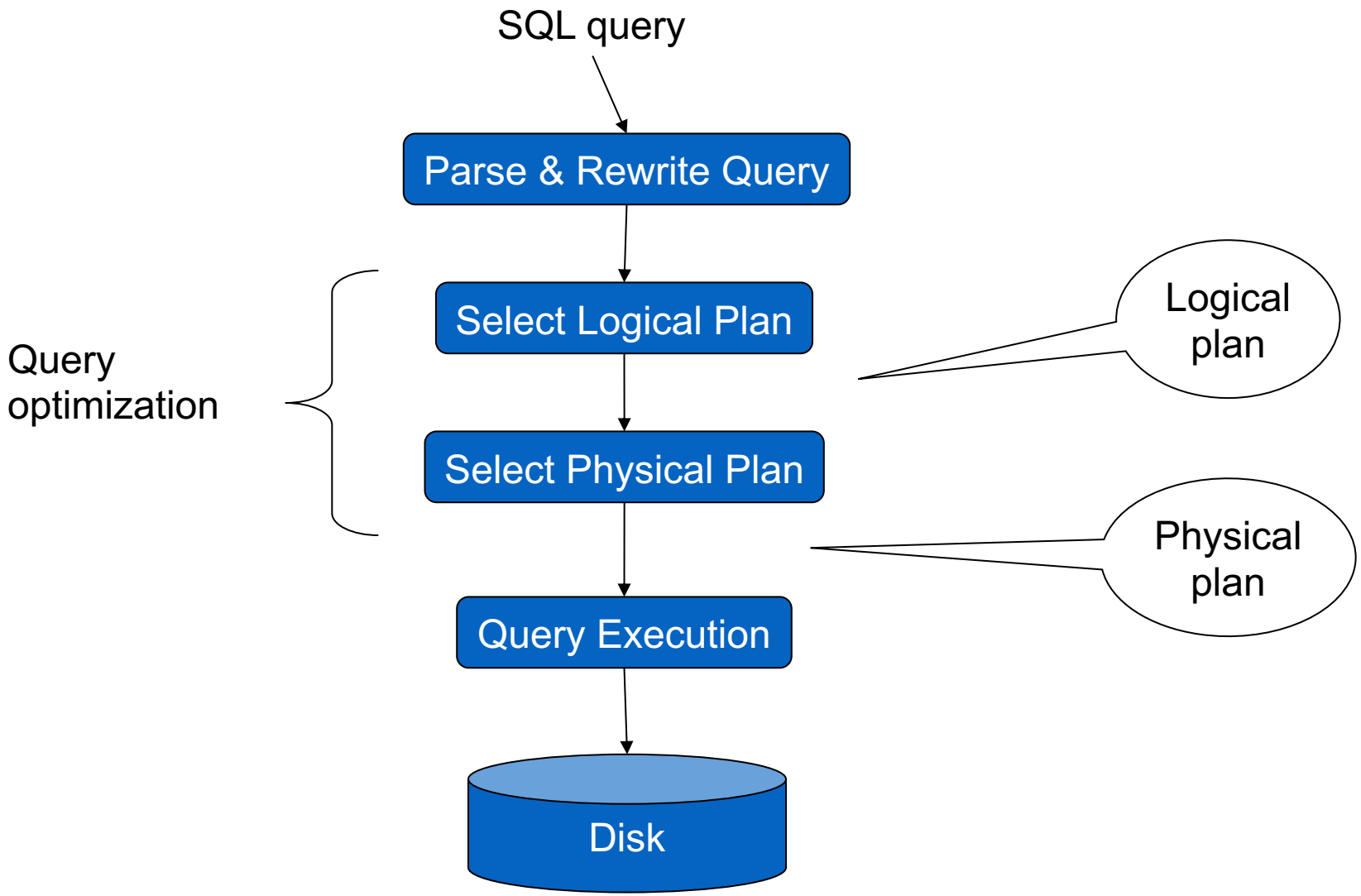
$$\pi_{sname}((Tempred \cap Tempgreen) \bowtie Sailors)$$

Equivalence to Relational Algebra

- The basic form of an SQL query is as follows:



Steps of the Query Processor



Steps in Query Evaluation

- Step 0: Admission control
  - User connects to the db with username, password
  - User sends query in text format
- Step 1: Query parsing
  - Parses query into an internal format
  - Performs various checks using catalog
    - Correctness, authorization, integrity constraints
- Step 2: Query rewrite
  - View rewriting, flattening, etc.

Rewritten Version of Our Query

- Original query:

```
SELECT sname
FROM NearbySupp
WHERE sno IN ( SELECT sno
               FROM Supplies
               WHERE pno = 2 )
```

- Rewritten query:

```
SELECT S.sname
FROM Supplier S, Supplies U
WHERE S.scity='Seattle' AND S.sstate='WA’
AND S.sno = U.sno
AND U.pno = 2;
```

Continue with Query Evaluation

- Step 3: Query optimization
  - Find an efficient query plan for executing the query

- A query plan is
  - Logical query plan: an extended relational algebra tree
  - Physical query plan: with additional annotations at each node
    - Access method to use for each relation
    - Implementation to use for each relational operator

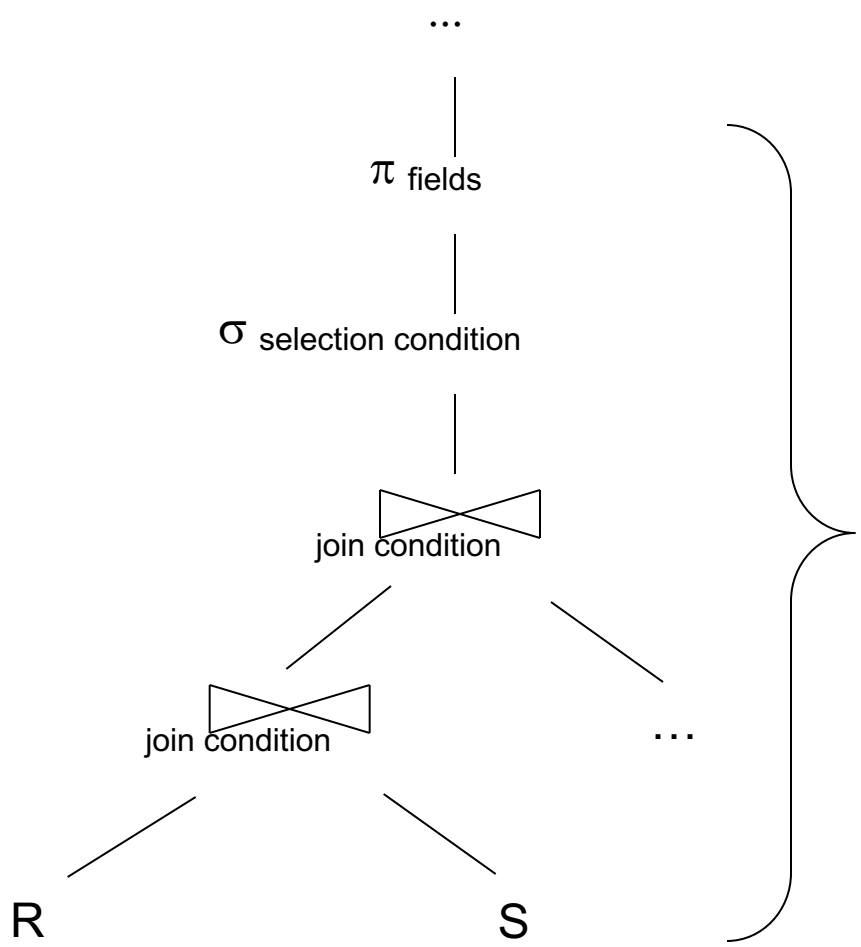
Extended Algebra Operators

- Union $\cup$, intersection $\cap$, difference -
- Selection s
- Projection π
- Join $\bowtie$
- Duplicate elimination d
- Grouping and aggregation g
- Sorting t
- Rename ρ

Query Block

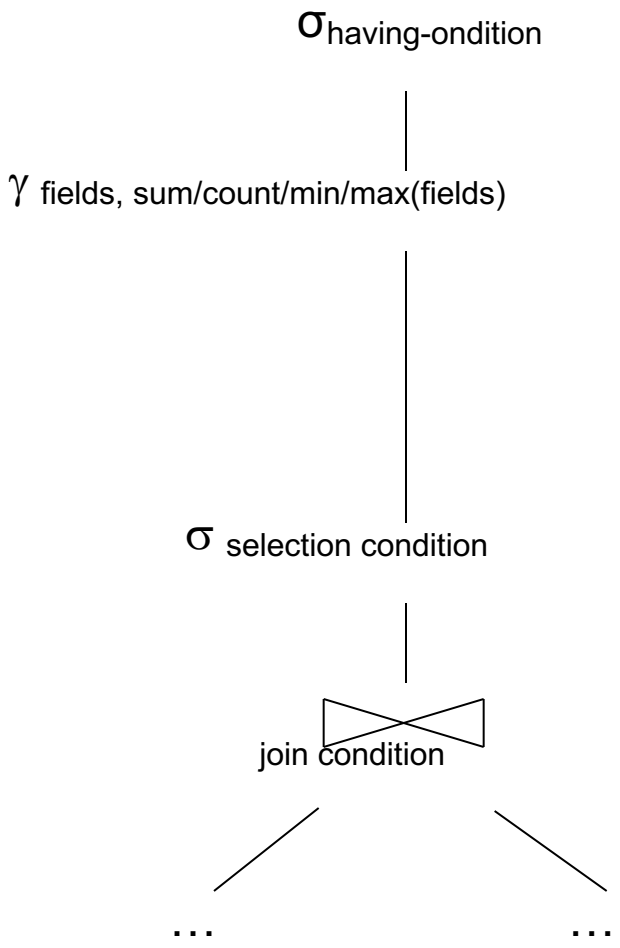
- Most optimizers operate on individual query blocks
- A query block is an SQL query with no nesting
  - Exactly one
    - SELECT clause
    - FROM clause
  - At most one
    - WHERE clause
    - GROUP BY clause
    - HAVING clause

Typical Plan for Block



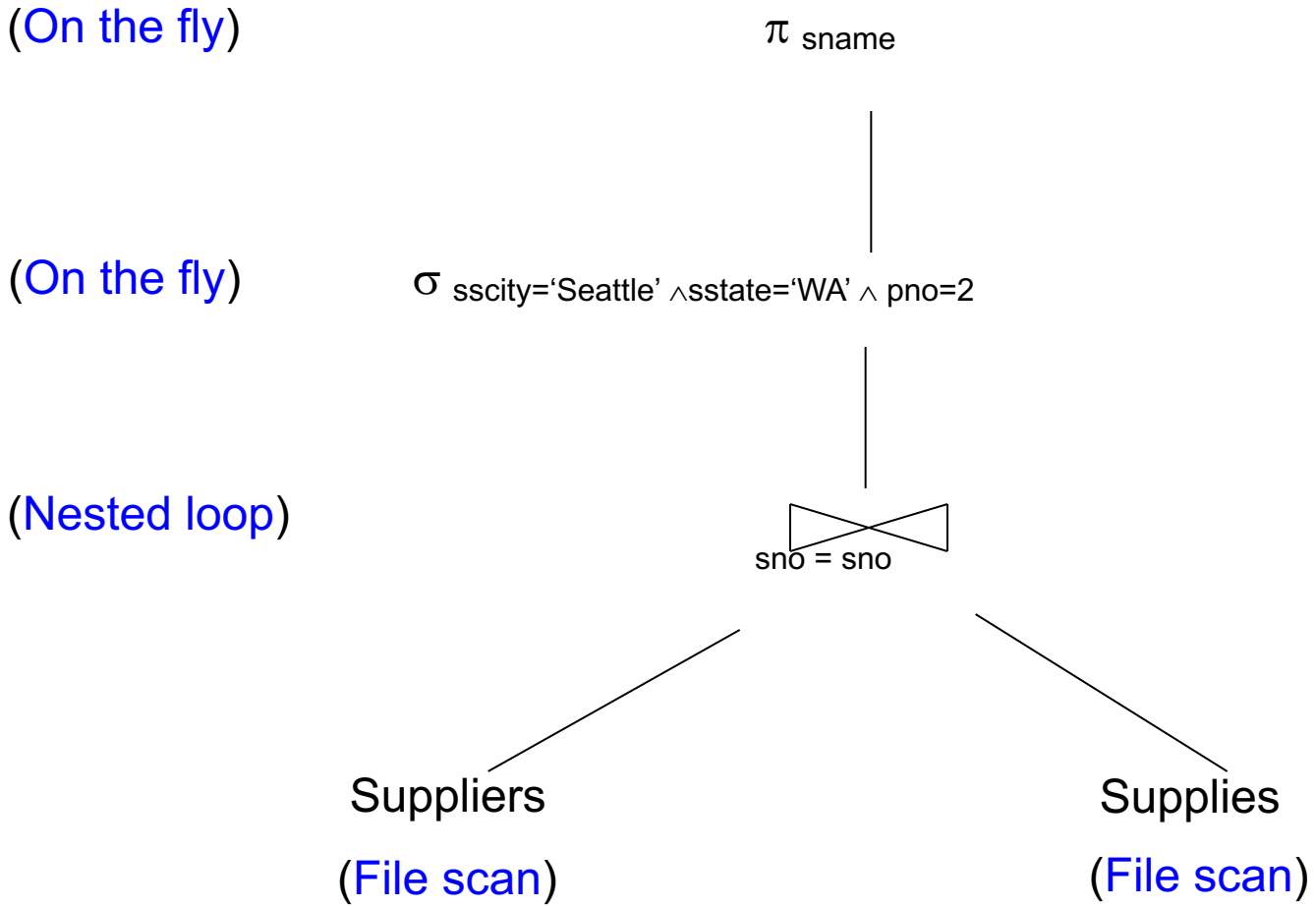
SELECT-PROJECT-JOIN
Query

Typical Plan For Block



Physical Query Plan

Supplier(sno,sname,scity,sstate)
Part(pno,pname,psize,pcolor)
Supply(sno,pno,price)



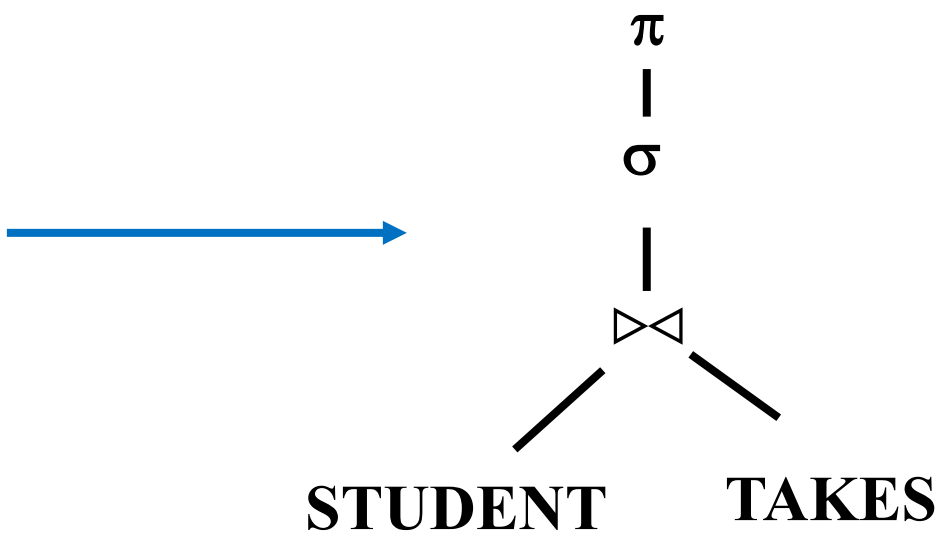
Final Step in Query Processing

- Step 4: Query execution
  - How to synchronize operators?
  - How to pass data between operators?

- What techniques are possible?
  - One thread per query
  - Iterator interface
  - Pipelined execution
  - Intermediate result materialization

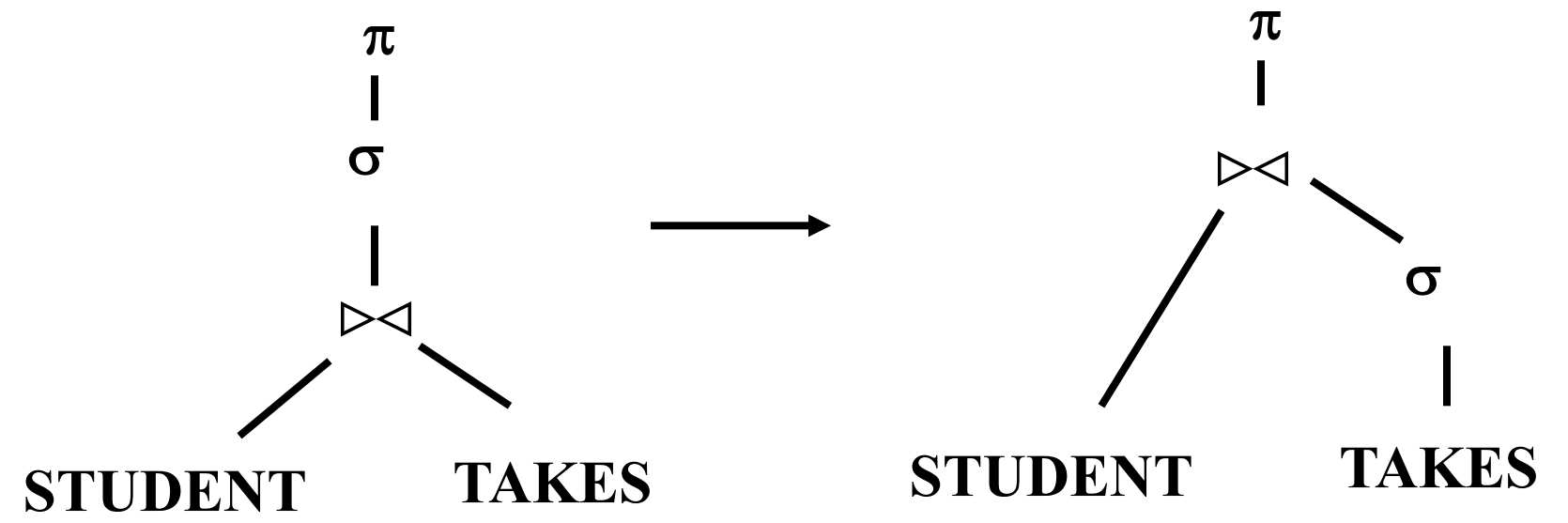
Translating SQL Queries Into Relational Algebra Trees

```
SELECT name
FROM STUDENT, TAKES
WHERE c-id='415' AND
STUDENT.ssn=TAKES.ssn
```



- An SQL block can be thought of as an algebra expression containing:
  - A cross-product of all relations in the FROM clause
  - Selections in the WHERE clause
  - Projections in the SELECT clause
- Remaining operators can be carried out on the result of such SQL block

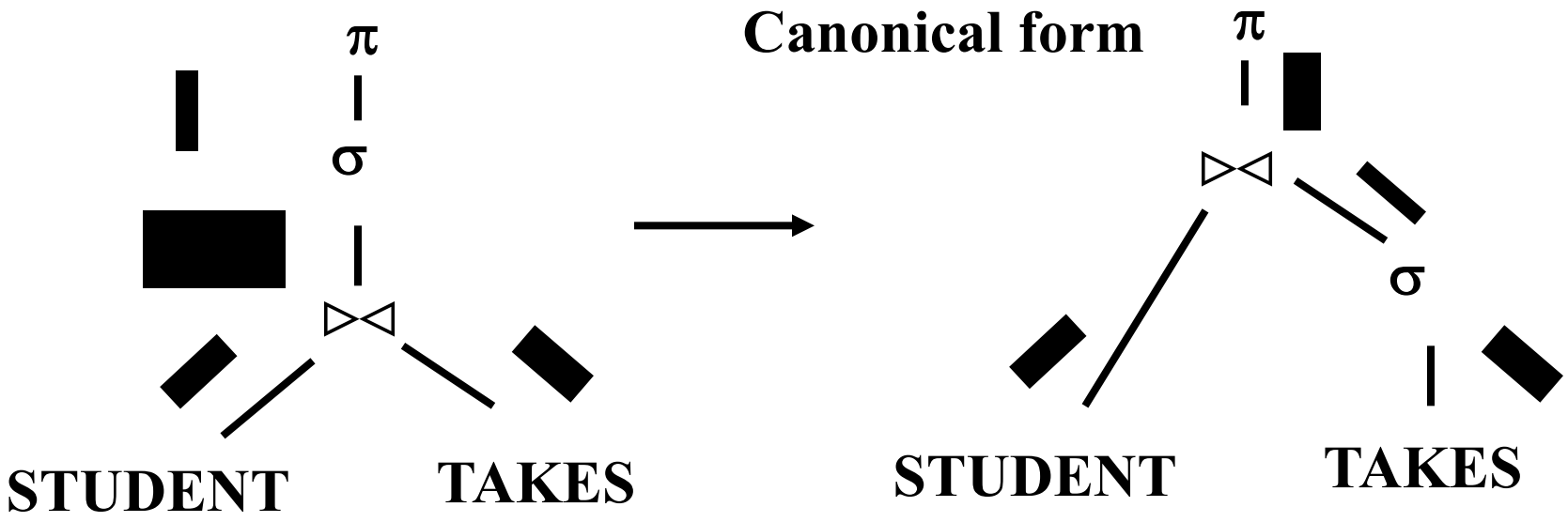
Translating SQL Queries Into Relational Algebra Trees



Still the same result!

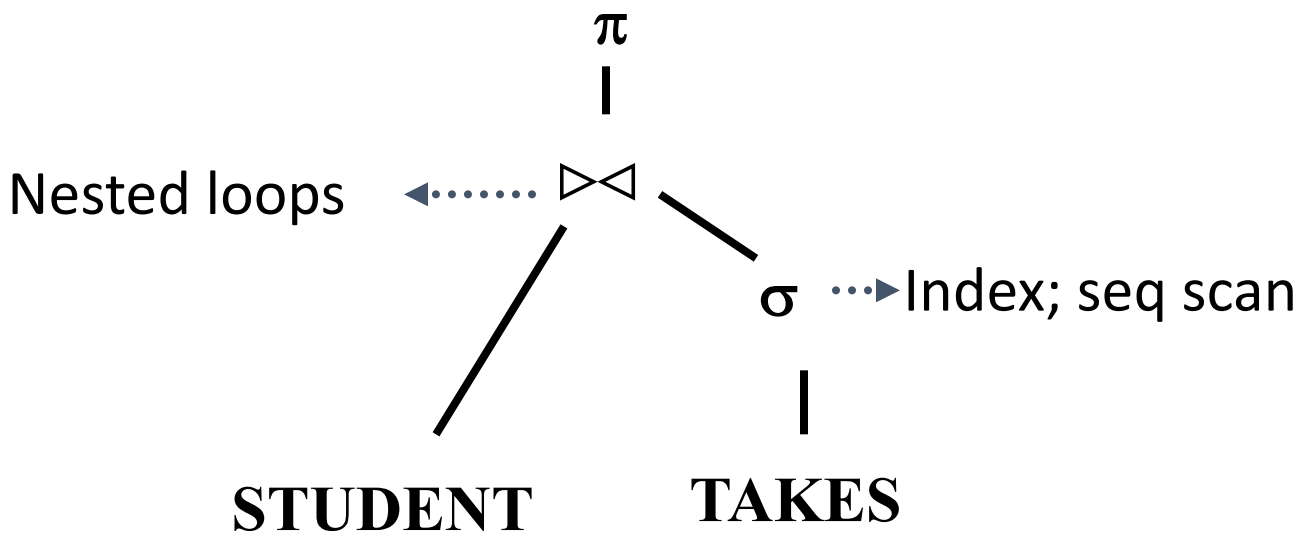
How can this be guaranteed?

Translating SQL Queries Into Relational Algebra Trees



OBSERVATION: try to perform selections and projections early!

Translating SQL Queries Into Relational Algebra Trees



How to evaluate a query plan (as opposed to evaluating an operator)?

Query Evaluation Plans

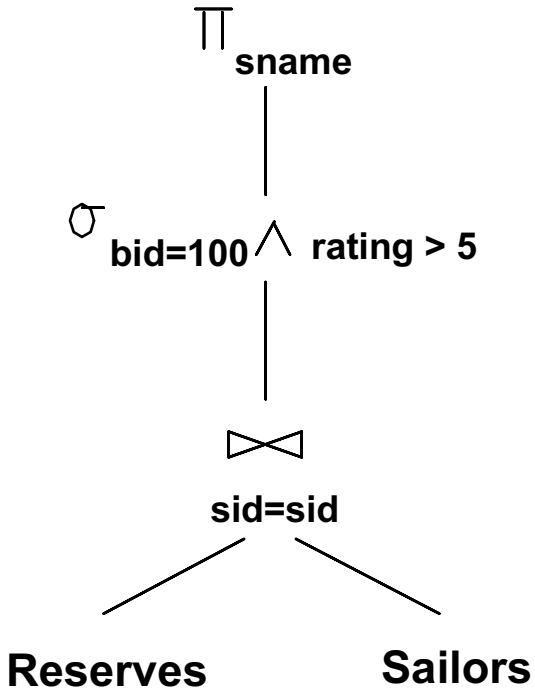
- A query evaluation plan (or simply a plan) consists of an extended relational algebra tree (or simply a tree)
- A plan tree consists of annotations at each node indicating:
  - The access methods to use for each relation
  - The implementation method to use for each operator
- Consider the following SQL query Q:

```
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid AND
   R.bid=100 AND S.rating>5
```

What is the corresponding RA of ***Q***?

Pushing Selections

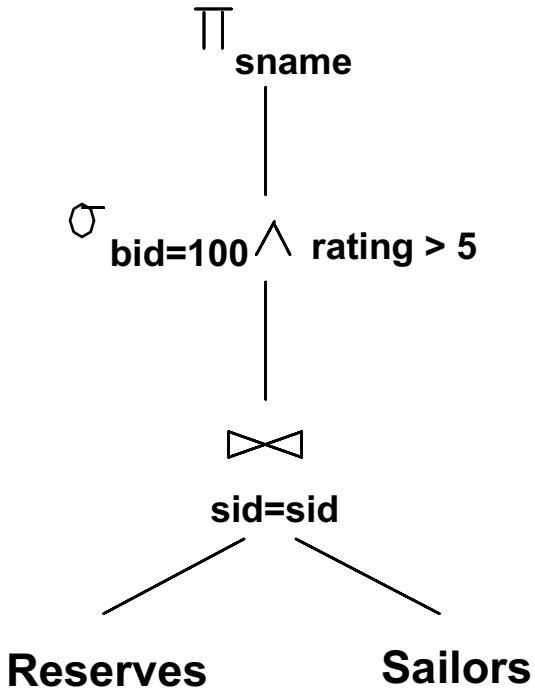
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid **AND**
R.bid=100 **AND** S.rating>5



Pushing Selections

```
SELECT  S.sname
FROM  Reserves R, Sailors S
WHERE  R.sid=S.sid AND
    R.bid=100 AND S.rating>5
```

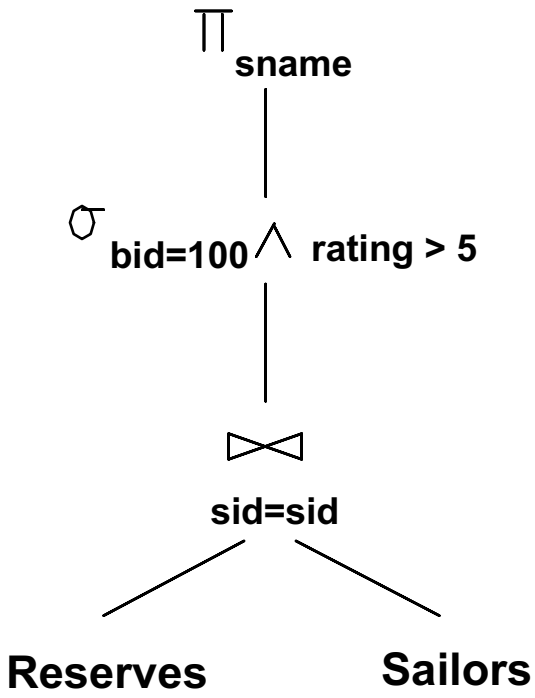
- How can we reduce the cost of a join?
  - By reducing the sizes of the input relations!

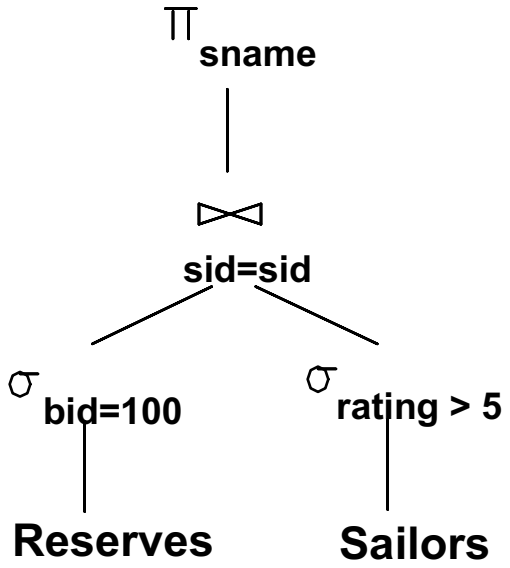


Pushing Selections

```
SELECT  S.sname
FROM  Reserves R, Sailors S
WHERE  R.sid=S.sid AND
    R.bid=100 AND S.rating>5
```

- How can we reduce the cost of a join?
  - By reducing the sizes of the input relations!





Relational Algebra Equivalences

- A relational query optimizer uses **relational algebra equivalences** to identify many **equivalent expressions** for a given query

- Two relational algebra expressions over the same set of input relations are said to be equivalent if they produce the same result on all relations' instances

- Relational algebra equivalences allow us to:
  - **Push selections and projections ahead of joins**
  - **Combine selections and cross-products into joins**
  - **Choose different join orders**

Another Example

country(name, code, capital)
city(name, country, population)
borders(country1, country2, length)

- Find the name of all countries that border Switzerland

Country

name	code	capital
Russia	1	Moscow
France	2	Paris
Spain	3	Madrid
Switzerland	4	Zurich
Italy	5	Rome
Germany	6	Berlin
…	..	…

borders

country1	country2
2	6
2	4
2	3
5	4
6	4
…	..

Another Example

country(name, code, capital)
city(name, country, population)
borders(country1, country2, length)

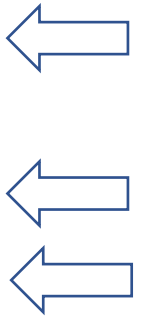
- Find the name of all countries that border Switzerland

Country

name	code	capital
Russia	1	Moscow
France	2	Paris
Spain	3	Madrid
Switzerland	4	Zurich
Italy	5	Rome
Germany	6	Berlin
…	..	…

borders

country1	country2
2	6
2	4
2	3
5	4
6	4
…	..



Another Example

country(name, <u>code</u>, capital)
city(<u>name</u>, <u>country</u>, population)
borders(<u>country1</u>, country2, length)

- Find the name of all countries that border Switzerland

Country

name	code	capital
Russia	1	Moscow
France	2	Paris
Spain	3	Madrid
Switzerland	4	Zurich
Italy	5	Rome
Germany	6	Berlin
…	..	…

$$\rho(s, \pi_{code}(\sigma_{name='Switzerland'}(country)))$$

s

code
4

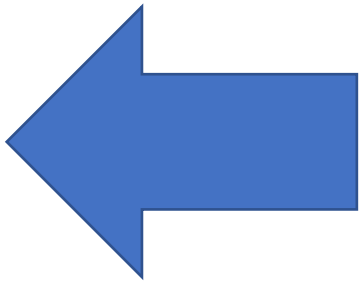
Another Example

country(name, <u>code</u>, capital)
city(<u>name</u>, <u>country</u>, population)
borders(<u>country1</u>, country2, length)

- Find the name of all countries that border Switzerland

temp1

country1	country2	code
2	6	4
2	4	4
2	3	4
5	4	4
6	4	4
…	..	….



$$\rho(s, \pi_{code}(\sigma_{name='Switzerland'}(country)))$$

$$\rho(b, borders)$$

$$\rho(temp1, \sigma_{b.country2=s.code}(b \times s))$$

Another Example

country(name, <u>code</u>, capital)
city(<u>name</u>, <u>country</u>, population)
borders(<u>country1</u>, country2, length)

- Solution 1

$$\rho(s, \pi_{code}(\sigma_{name='Switzerland'}(country)))$$
$$\rho(b, borders)$$
$$\rho(temp1, \sigma_{b.country2=s.code}(b \times s))$$
$$\pi_{name}(\sigma_{temp1.country1=country.code}(\pi_{country1}(temp1) \times country))$$

- Solution 2

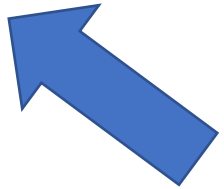
$$\rho(c', country)$$
$$\rho(c'', country)$$
$$\rho(b, border)$$
$$\rho(temp1, (\sigma_{c'.code=b.country1 \wedge c''.code=b.country1 \wedge c'.name='Switzerland'}(c' \times c'' \times b))$$
$$\pi_{c''.name}(temp1)$$

Another Example

country(name, <u>code</u>, capital)
city(<u>name</u>, <u>country</u>, population)
borders(<u>country1</u>, country2, length)

- Find the name of all countries that border Switzerland

country1	name	capital
2	France	Paris
5	Italy	Rome
6	Germany	Berlin
…	..	….



$$\rho(s, \pi_{code}(\sigma_{name='Switzerland'}(country)))$$

$$\rho(b, borders)$$

$$\rho(temp1, \sigma_{b.country2=s.code}(b \times s))$$

$$\pi_{name}(\sigma_{temp1.country1=country.code}(\pi_{country1}(temp1) \times country))$$