CS 5530

Database Systems
Spring 2020

Relational Algebra and SQL

Project Phase 1

- HW2 solution released
- Make sure you don’t make the same errors on Phase 1

Ponder…

- Consider a database for cars and parts

PartsUsedIn

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

Cars

cNo	Name
c1	Delorian
c2	Pinto
c3	Prius
c4	Ram

Parts

pNo	Name
p1	Spark Plug
p2	Battery
p3	Flux Capacitor
p4	Hitch

Ponder…

- Get all cars that use part `p1`

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

$$\pi_{cNo}(\sigma_{pNo='p1'}(A))$$

Ponder…

- Get all cars that use parts `p2` **and** `p4`

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

Ponder…

- Get all cars that use parts `p2` and `p4`

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

pNo
p2
p4

First, make a new table containing parts we are interested in

Division

- Get all cars that use parts `p2` and `p4`

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

pNo
p2
p4

A / B

cNo
c1
c4

Introduce new operator: /

Division

- Get all cars that use parts `p2` and `p4`

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

pNo
p2
p4

A / B

cNo
c1
c4

Introduce new operator: /

schema(B) is subset of schema(A)

schema(A / B) is subset of schema(A)

Division

- Get all cars that use various sets of parts…

A

cNo	pNo
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

pNo
p2
p4

A / B

cNo
c1
c4

C

pNo
p2

A / C

cNo
c1
c2
c3
c4

D

pNo
p1
p2
p4

A / D

cNo
c1

Division

- Compose division from π σ $\times$ — $\cup$

A

x	y
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

y
p2
p4

A / B

x
c1
c4

find all x s.t. $\forall y \in B \ \exists (x,y) \in A$

Division

- Compose division from π σ $\times$ — $\cup$

A

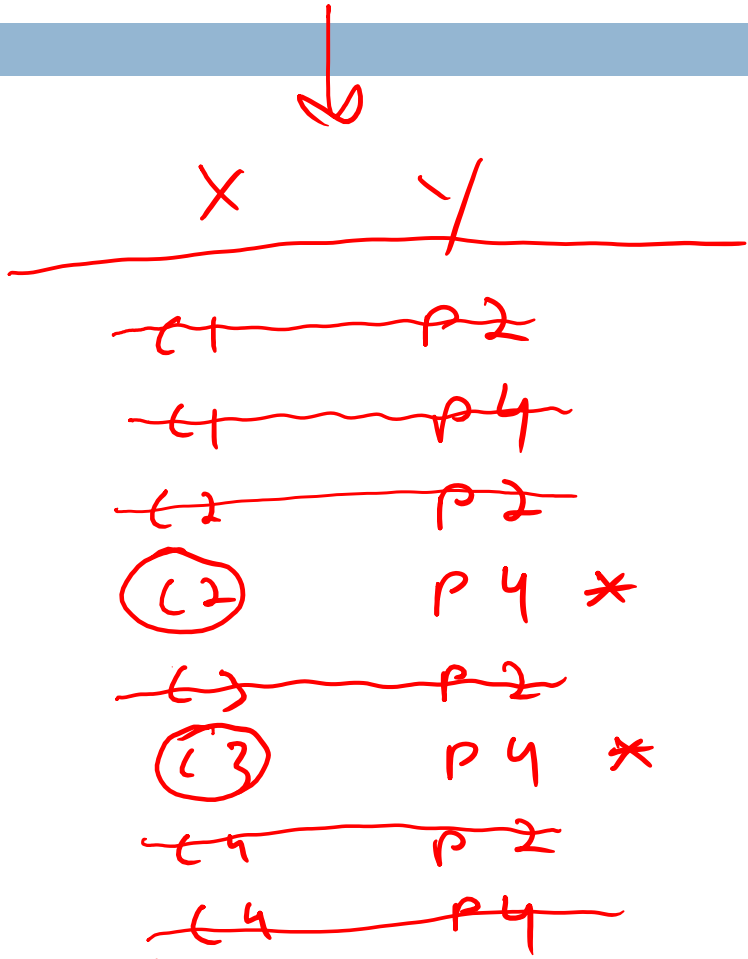
x	y
c1	p1
c1	p2
c1	p3
c1	p4
c2	p1
c2	p2
c3	p2
c4	p2
c4	p4

B

y
p2
p4

A / B

x
c1
c4



$\pi_x(A) - \pi_x\left(\left(\pi_x(A) \times B\right) - A\right)$

Natural Join

- R1 ⋈ R2 → R3
- Just like SQL natural join
- Select all rows from R1 × R2 where their common attributes have equal values
- Output schema has just 1 copy of common attributes

Theta Join

- R1 $\bowtie_{\text{condition}}$ R2 → R3
- Like SQL `JOIN … WHERE`
- Output schema same as R1 × R2

Theta Join

- R1 $\bowtie_{\text{condition}}$ R2 → R3
- Like SQL `JOIN … WHERE`
- Output schema same as R1 × R2
- Shortcut for $\sigma_{\text{condition}}$(R1 × R2)

Rename

- Often useful to declare ‘variables’ by renaming a relation
- ρ(newname, expression)

Rename

- Often useful to declare 'variables' by renaming a relation
- ρ(newname, expression)
- Example:

$\rho(\text{HerbertSerials}, \pi_{\text{Serial}}(\sigma_{\text{Author=Herbert}}(\text{Titles} \bowtie \text{Inventory})))$

Inventory

Serial	ISBN
1001	978-0441172719
1002	978-0441172720

Titles

ISBN	Title	Author
978-0441172719	Dune	Herbert
978-0441172720	Children of Dune	Herbert

Rename

- Often useful to declare ‘variables’ by renaming a relation
- ρ(newname, expression)
- Example:

$\rho(\text{HerbertSerials}, \pi_{\text{Serial}}(\sigma_{\text{Author=Herbert}}(\text{Titles} \bowtie \text{Inventory})))$

- Now we can reuse HerbertSerials

$\pi_{\text{CardNum}}(\text{HerbertSerials} \bowtie \text{CheckedOut})$

Example

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

Example

- Names of Students who earned an A or B

$$\rho(TempIDs, \pi_{sID}(\sigma_{Grd=A \text{ || } Grd=B}(Enrolled)))$$

$$\pi_{Name}(TempIDs \bowtie Students)$$

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

Example

- Names of Students who earned an A *and* B

$$\rho(As, \pi_{sID}(\sigma_{Grd=A}(Enrolled)))$$

$$\rho(Bs, \ldots B \ldots)$$

$$\pi_{Name}((As \cap Bs) \bowtie Students)$$

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

Exercise

1. Names of Courses with a student born in 1979 or a student born in 1982

2. Names of Students taking *all Courses*
   - Hint: consider the divide operator

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

Exercise

Names of Courses with a student born in 1979 or a student born in 1982

$\rho(sIDs, \pi_{sID}(\sigma(DOB = 79 \vee DOB = 82(Students))$

$\pi_{Name}(\pi_{cID}(sIDs \bowtie Enrolled) \bowtie Courses)$

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

Exercise

Names of Students taking *all Courses*

$\rho(sIDs, \pi_{sID,cID}(Enrolled) / \pi_{cID}(Courses))$

$\pi_{Name}(sIDs \bowtie Students)$

Students

sID	Name	DOB
1	Hermione	1980
2	Harry	1979
3	Ron	1980
4	Malfoy	1982

Enrolled

sID	cID	Grd
1	3500	A
1	3810	A-
1	5530	A
2	3810	A
2	5530	B

Courses

cID	Name
3500	SW Practice
3810	Architecture
5530	Databases

RA → SQL

- Basic translations from RA → SQL
  - I can't go over every possibility
  - You'll have to reason about it sometimes…

Project(π) → SELECT

- `SELECT col1, col2, …`
  - If the RA query projects $\pi_{col1, col2, \ldots}$
- `SELECT *`
  - If the RA query has no projection π

Select(σ) → `WHERE`

- `WHERE <condition>`
  - If the RA query has selection $\sigma_{\text{<condition>}}$

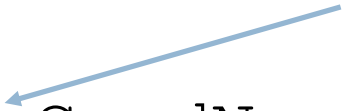
Rename(ρ)

- We can rename relations within a query

```
select p.CardNum
from Patrons p
where p.Name = 'Joe';
```

Rename(ρ)

- We can rename relations within a query



Can use it *before* its declaration!

```
select p.CardNum
from Patrons p
where p.Name = 'Joe';
```

Nested Queries

- Give a name to a temp query
- Find Serials of “The Lorax”

```
select Serial from
   (select ISBN from Titles where
     Title = 'The Lorax') as lorax
natural join Inventory;
```

Nested Queries

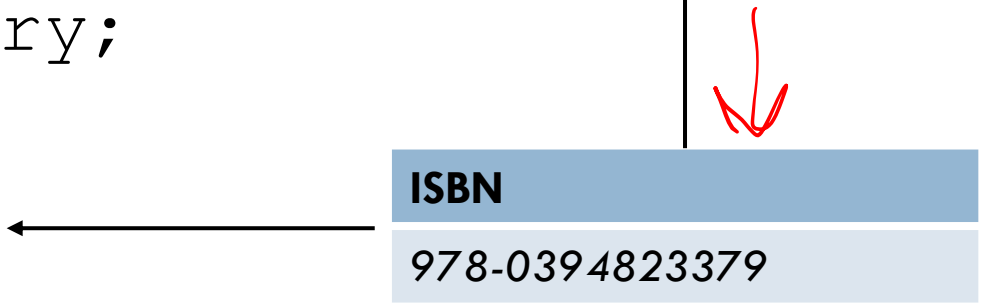
- Give a name to a temp query
- Find Serials of "The Lorax"

```
select Serial from

   (select ISBN from Titles where
     Title = 'The Lorax') as lorax

natural join Inventory;
```

Serial	ISBN
…	…
1004	*978-0394823379*
1005	*978-0394823379*
…	…



Cross Product → JOIN

Relational Algebra	SQL
$R1 \times R2$	`R1 JOIN R2`
$R1 \bowtie R2$	`R1 NATURAL JOIN R2`
$R1 \bowtie_{condition} R2$	`R1 JOIN R2 WHERE …`

Union(∪) → UNION

- Find all locations

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

Union(∪) → UNION

- Find all locations

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

- π_{Addr}(CorporateLocs ∪ RetailLocs)

Union(∪) → UNION

- Find all locations

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

- π_{Addr}(CorporateLocs ∪ RetailLocs)

```
SELECT Addr FROM CorporateLocs
UNION
SELECT Addr FROM RetailLocs
```

Union(∪) → UNION

- Find all locations

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

- π_{Addr}(CorporateLocs ∪ RetailLocs)

```
SELECT Addr FROM CorporateLocs
UNION
SELECT Addr FROM RetailLocs
```

Two SELECTs required

UNION ALL

- Sometimes we don't care about duplicates

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

```
SELECT Addr FROM CorporateLocs
UNION ALL
SELECT Addr FROM RetailLocs
```

Addr
400s State St.
750 Rose Park
455 Pine Rd.
455 Pine Rd.
123 Fake St.
50 S. Campus

Intersect(∩) → INTERSECT

- Find locations with corporate and retail

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

- $\pi_{Addr}(CorporateLocs) \cap \pi_{Addr}(RetailLocs)$

```
SELECT Addr FROM CorporateLocs
INTERSECT
SELECT Addr FROM RetailLocs
```

Intersect(∩) → INTERSECT

- Find locations with corporate and retail

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

```
SELECT Addr FROM CorporateLocs
INTERSECT
SELECT Addr FROM RetailLocs
```

- …except MySQL doesn't support it

Intersect in MySQL

- How can we formulate intersect?

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

Intersect in MySQL

- Lots of ways to formulate it

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

```
SELECT * FROM
(SELECT Addr FROM CorporateLocs) AS corp
NATURAL JOIN
(SELECT Addr FROM RetailLocs) AS retail;
```

Intersect in MySQL

- Lots of ways to formulate it

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

```
SELECT * FROM
(SELECT Addr FROM CorporateLocs) AS corp
NATURAL JOIN
(SELECT Addr FROM RetailLocs) AS retail;
```

- If schemas are the same, natural join = intersect

IN

- Filter by nested query

```
SELECT x FROM … WHERE y IN
(SELECT …);
```

IN

- Filter by nested query

```
SELECT x FROM … WHERE y IN
(SELECT …);
```

No ‘as’

- If nested query appears after `IN`, it is not renamed

IN

- Filter by nested query

```
SELECT x FROM … WHERE y IN
(SELECT …);
```

  - Nested query must have same column type(s) as `x`

- Any `x` that are part of nested query are returned

Intersect in MySQL

- Using `IN`

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

```
SELECT Addr FROM CorporateLocs
WHERE
Addr IN (SELECT Addr FROM RetailLocs);
```

Set Difference(—)

- Find all locations that are corporate-only

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

$\pi_{Addr}(CorporateLocs) - \pi_{Addr}(RetailLocs)$

Set Difference(—) → NOT IN

- Find all locations that are corporate-only

CorporateLocs

MngrID	Addr
1	455 Pine Rd.
2	123 Fake St.
5	50 S. Campus

RetailLocs

MngrID	Addr
6	400s State St.
3	750 Rose Park
4	455 Pine Rd.

$\pi_{Addr}(CorporateLocs) - \pi_{Addr}(RetailLocs)$

```
SELECT Addr FROM CorporateLocs
WHERE
Addr NOT IN (SELECT Addr FROM RetailLocs);
```

Exercise

Patrons

Name	CardNum
Joe	1
Ann	2
Ben	3
Dan	4

Inventory

Serial	ISBN
1001	978-0590353427
1002	978-0590353427
1003	978-0679732242
1004	978-0394823379

CheckedOut

CardNum	Serial
1	1001
1	1004
4	1005

Phones

CardNum	Phone
1	555-5555
2	666-6666

1. All Patrons who have not checked out a book
2. All Patrons who have checked out 'The Lorax' AND 'Harry Potter'

ISBN	Title	Author
978-0590353427	Harry Potter	Rowling
978-0679732242	The Sound and the Fury	Faulkner
978-0394823379	The Lorax	Seuss
978-0062278791	Profiles in Courage	Kennedy
978-0441172719	Dune	Herbert