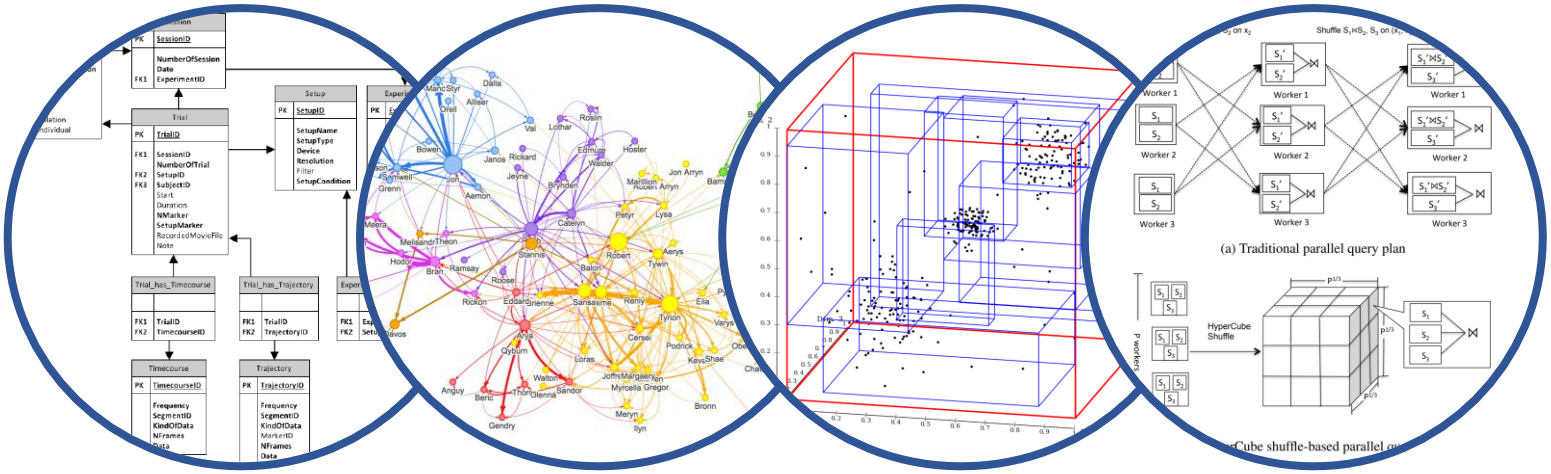


Introduction to Data Management

What did it cost? Tuples.

Shana Hutchison

Paul G. Allen School of Computer Science and Engineering
University of Washington, Seattle

Hw1 Feedback

- Success stories
  - "I learned that **my teammates can see my blind spots**"
  - "Although this assignment wasn't too complicated, I think trying out **pair-programming** was a positive experience, because I left with a greater understanding of both the syntax of SQL and the big-picture relational algebra flow required to achieve the outcomes desired."
  - "…The collaborative process allowed me to **run logic through another person** and come to conclusions at a fairly efficient rate; thus, I'm grateful for having had this opportunity."
  - "…it is just nice **knowing someone else in the class**. It can be intimidating in these large classes to put myself out there and get to know others so having an easy connection like this is nice."
  - "I like a lot that X was **audibly saying what X was doing while X typed** so I still felt included. It made it easier for me to stay engaged and think about what we were doing and continue to contribute."
- Many comments on learning git / collaborating through git

Hw1 Feedback

- Reminder on pair programming
  - Roles: *Coder*, *Navigator* on one computer
  - Try not to "split up" the assignment
- Meet in person!
  - "I would prefer if we worked side by side"
  - "On future assignments, I would like to work in person."
- Consider your partner's needs
  - Does your partner live off-campus?
  - Try meeting during the day, or use Skype/G Hangouts
- Anonymous feedback is anonymous
  - I don't know who you or your partner are!

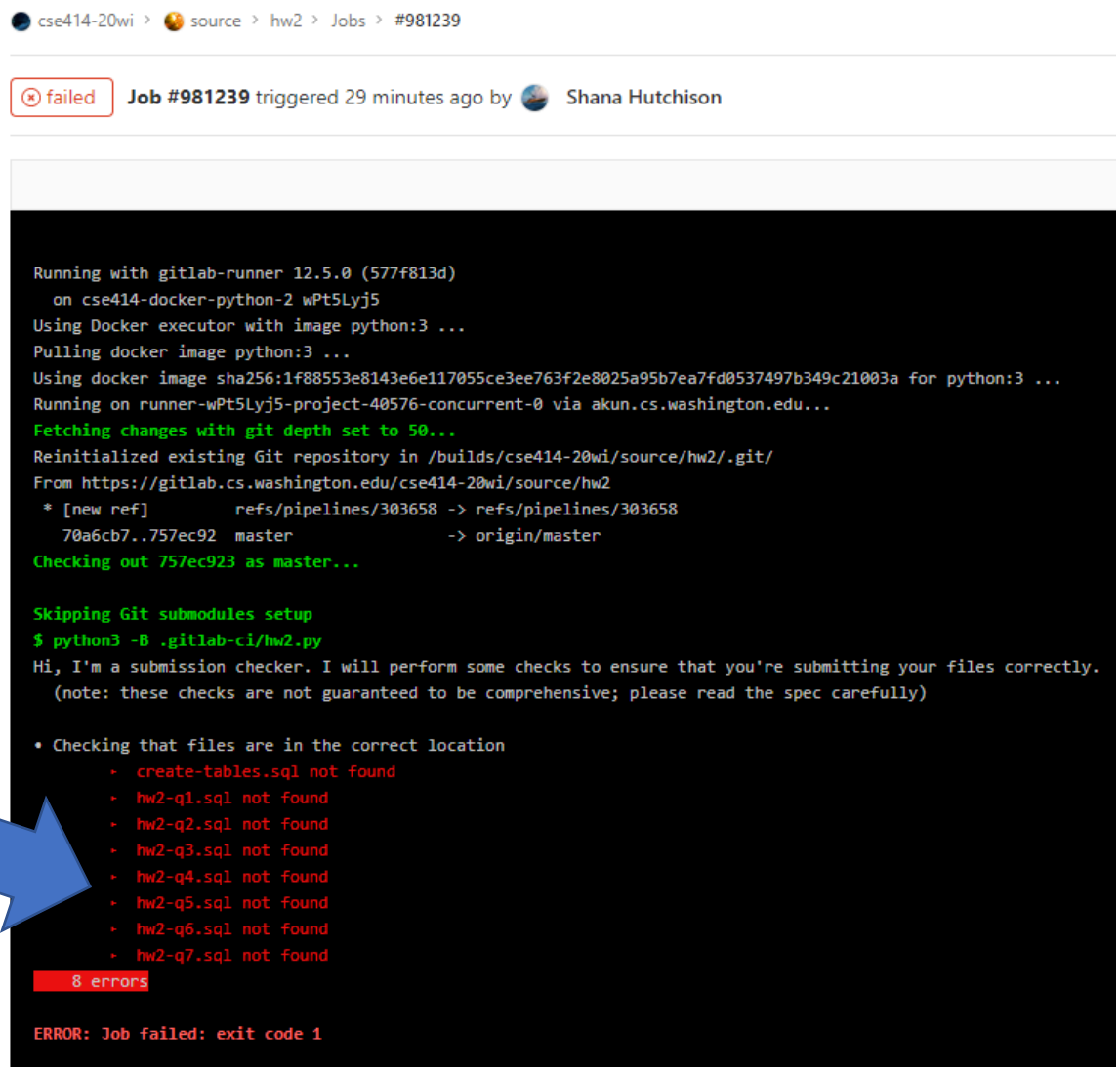
Hw2 CI

Continuous integration testing

- Runs after every push!
- Emails you if there's a failure

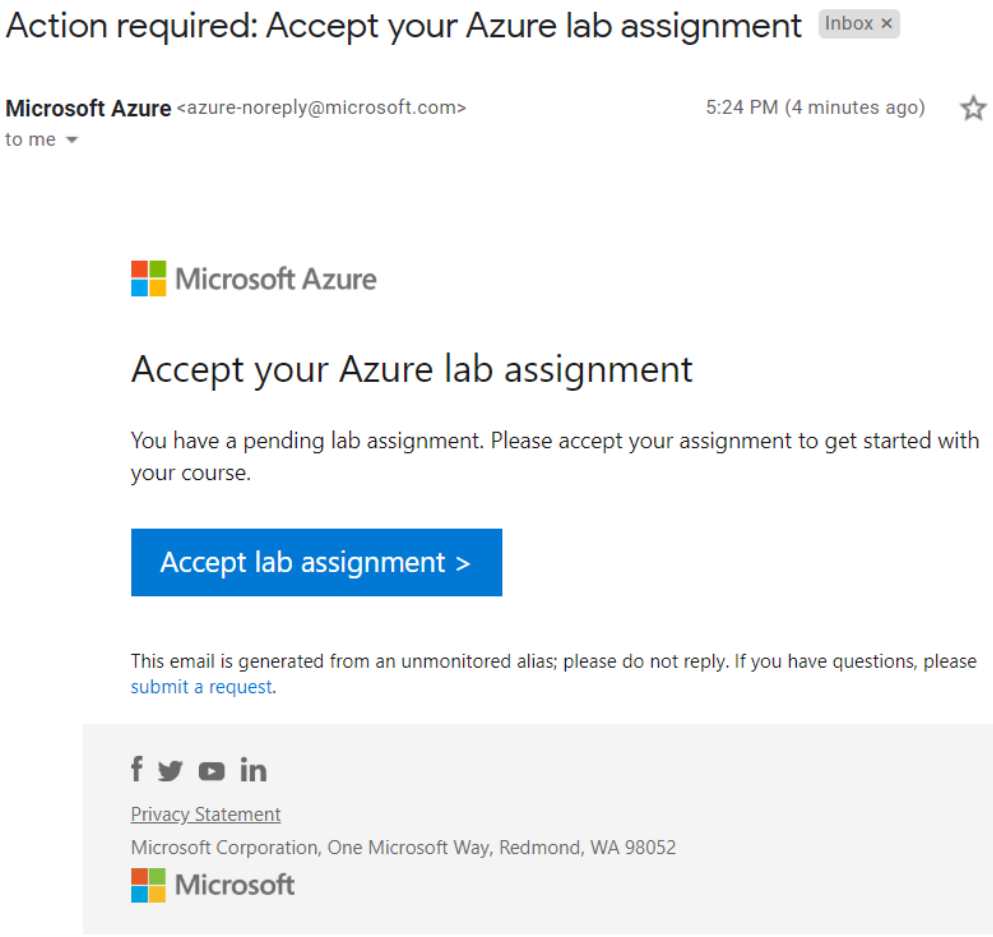
- Here, ci checker cannot find my submission files
  - I didn't write them yet!

Checks are super basic for hw2



Hw3 Check

- Hw3 – did you get your lab invite?



Practice

- Google "SQL practice problems"
- https://www.w3resource.com/sql-exercises/subqueries/index.php

Emp_details

EMP_IDNO	EMP_FNAME	EMP_LNAME	EMP_DEPT
127323	Michale	Robbin	57
526689	Carlos	Snares	63
843795	Enric	Dosio	57
328717	Jhon	Snares	63
444527	Joseph	Dosni	47
659831	Zanifer		
847674	Kuleswar		
748681	Henrey		
555935	Alex		
539569	George		
733843	Mario		
631548	Alan		
839139	Maria		

Emp_department

DPT_CODE	DPT_NAME	DPT_ALLOTMENT
57	IT	65000
63	Finance	15000
47	HR	240000
27	RD	55000
89	QC	75000

Practice 1

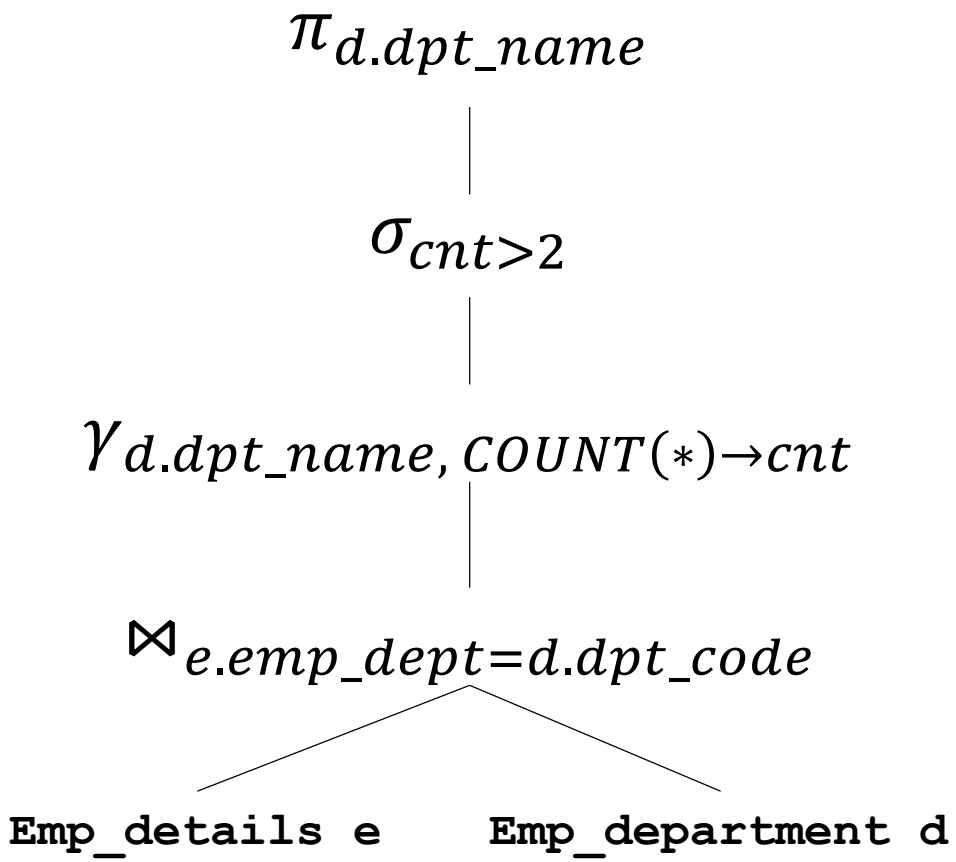
```
Emp_department(dpt_code, dpt_name, dpt_allotment,
               PRIMARY KEY (dpt_code));

Emp_details(emp_idno, emp_fname, emp_lname, emp_dept,
        PRIMARY KEY (emp_idno),
        FOREIGN KEY (emp_dept) REFERENCES Emp_department);
```

- Q: Find the *names of departments* where *more than two employees* are working.

  - Write a SQL query
  - Draw an RA plan

Practice 1

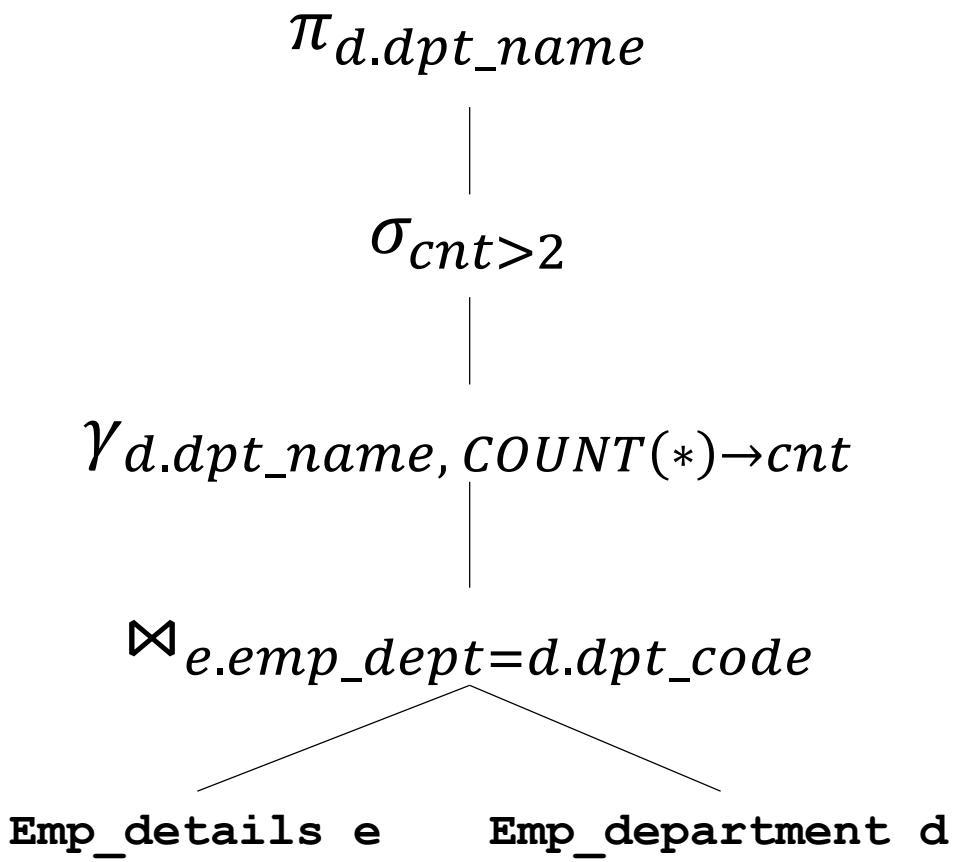
Find the *names of departments* where *more than two employees* are working.



Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 1

Find the *names of departments* where *more than two employees* are working.



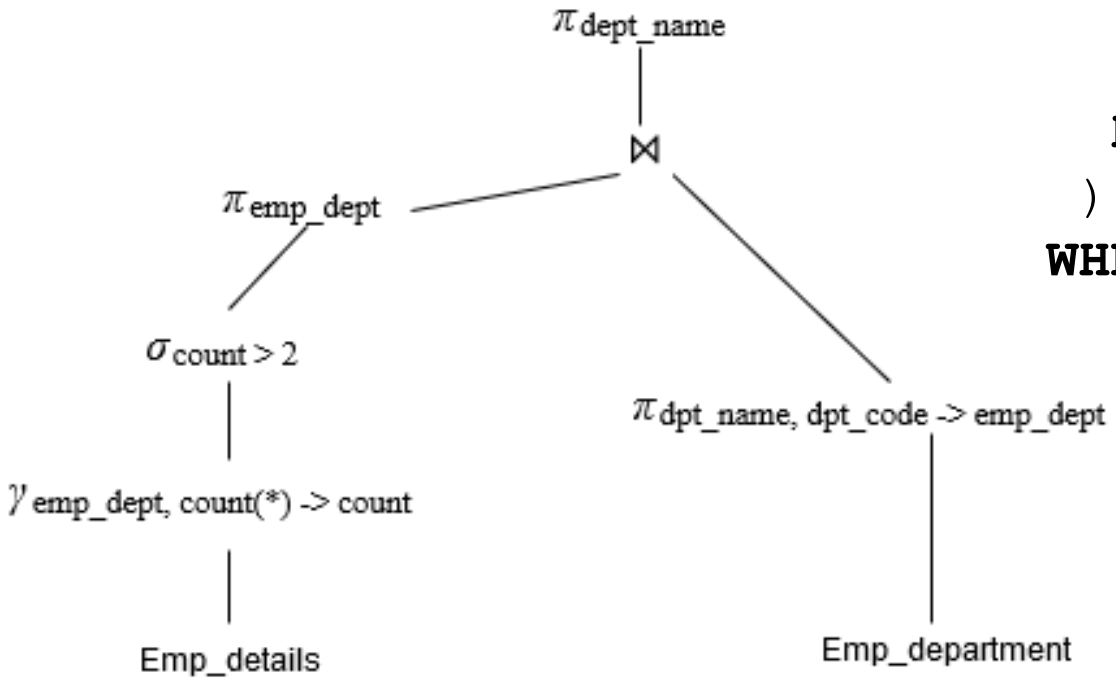

```
SELECT d.dpt_name
  FROM Emp_details e,
       Emp_department d
 WHERE e.emp_dept
       = d.dpt_code
 GROUP BY d.dpt_name
HAVING COUNT(*) > 2
```

Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 1

Find the *names of departments* where *more than two employees* are working.

- Another solution!
- Can you verify equivalence to the first?




```
SELECT d.dpt_name
  FROM (
   SELECT emp_dept
     FROM Emp_details
    GROUP BY emp_dept
   HAVING COUNT(*) > 2
  ) e, Emp_department d
 WHERE e.emp_dept
       = d.dpt_code
```

Practice 2

```
Emp_department(dpt_code, dpt_name, dpt_allotment,
               PRIMARY KEY (dpt_code));

Emp_details(emp_idno, emp_fname, emp_lname, emp_dept,
        PRIMARY KEY (emp_idno),
        FOREIGN KEY (emp_dept) REFERENCES Emp_department);
```

- Q: Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allotment is *second lowest*.
  - Write a SQL query
  - Draw an RA plan

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allotment is *second lowest*.

Let's take this in stages.

1. Find lowest allotment

```
SELECT min(d3.dpt_allotment)
FROM emp_department d3
```

Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allotment is *second lowest*.

Let's take this in stages.

1. Find lowest allotment
2. Find second lowest allotment

```
SELECT MIN(d2.dpt_allotment)
  FROM emp_department d2
 WHERE d2.dpt_allotment > (
       SELECT min(d3.dpt_allotment)
       FROM emp_department d3)
```

Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allotment is *second lowest*.

```
SELECT e.emp_fname, e.emp_lname
  FROM emp_details e, emp_department d
 WHERE e.emp_dept = d.dpt_code
   AND d.dpt_allotment = (
       SELECT MIN(d2.dpt_allotment)
         FROM emp_department d2
        WHERE d2.dpt_allotment > (
              SELECT min(d3.dpt_allotment)
              FROM emp_department d3));
```

Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allotment is *second lowest*.

Let's take this in stages.

1. Find lowest allotment

$$\gamma_{MIN(alloc)\rightarrow alloc}$$

|

Emp_department d

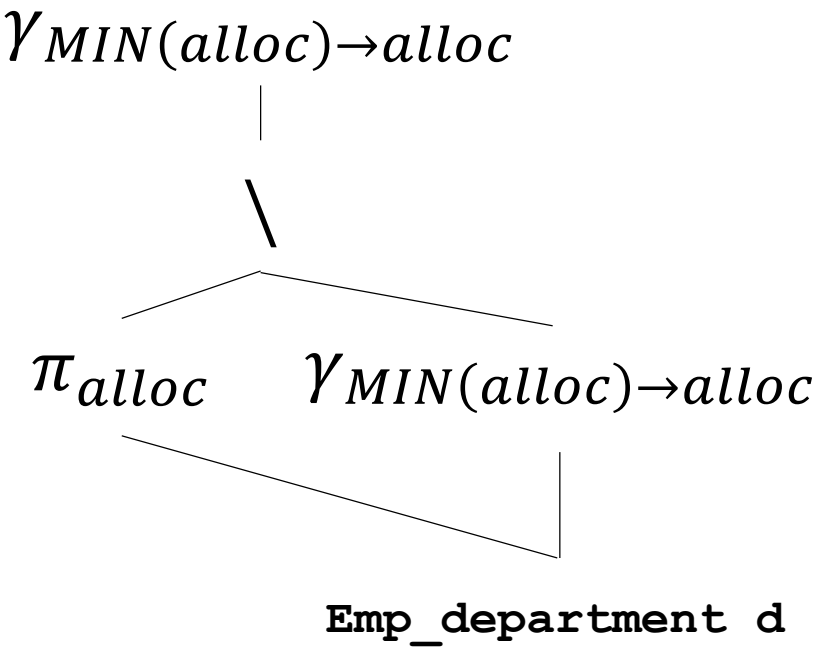
Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allocation is *second lowest*.

Let's take this in stages.

1. Find lowest allotment
2. Find second lowest allotment

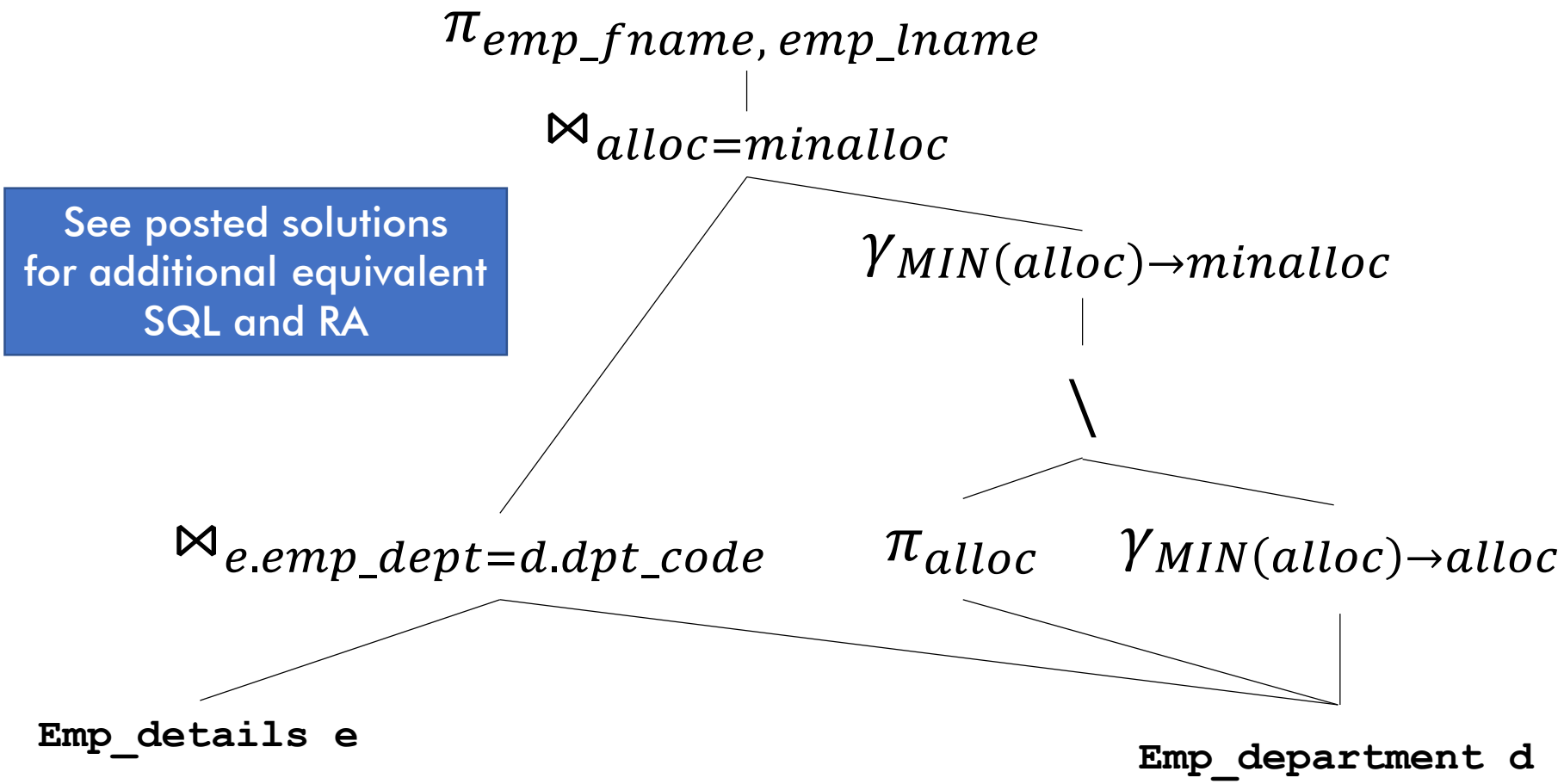


Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Practice 2

Write a query in SQL to find the *first name* and *last name* of employees working for departments whose allocation is *second lowest*.

See posted solutions for additional equivalent SQL and RA



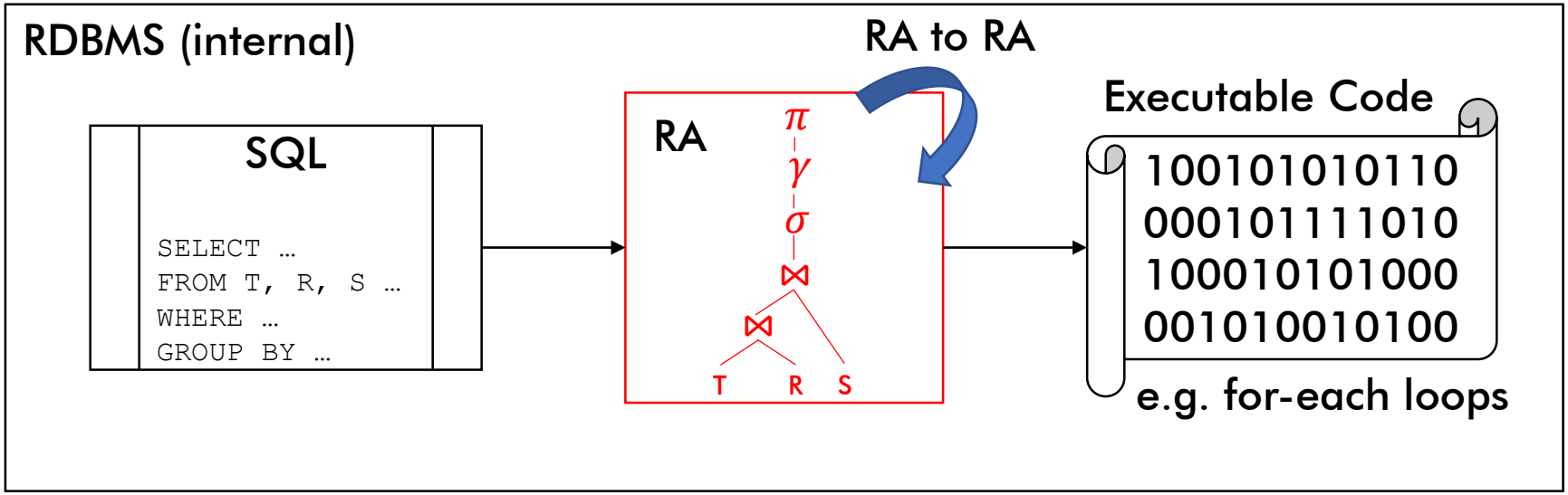
Emp_department(dpt_code, dpt_name, dpt_allotment)
Emp_details(emp_idno, emp_fname, emp_lname, emp_dept)

Hey, What's the Point of RA?

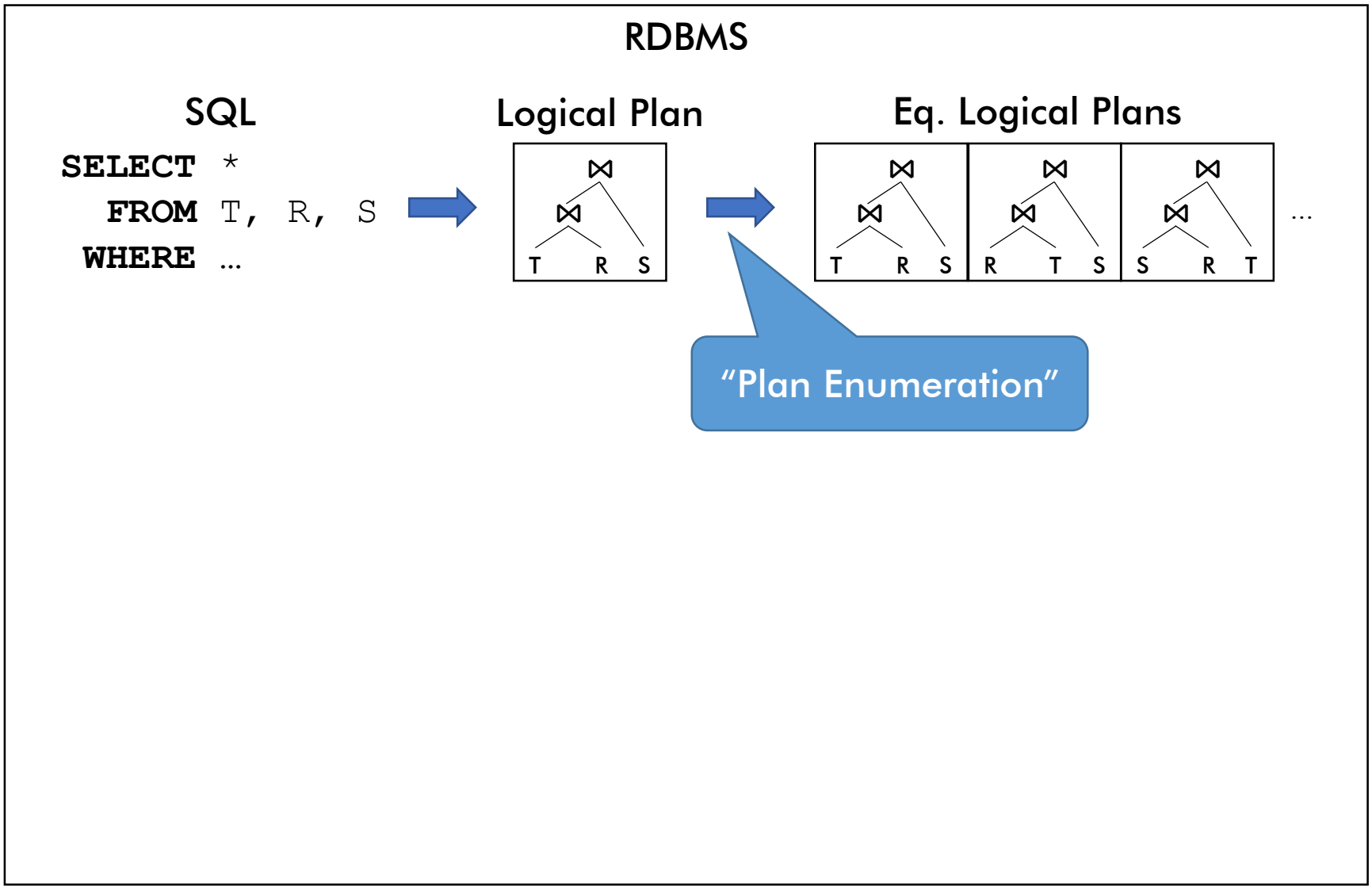
Overview of query optimization

1. RDBMS converts SQL to RA
2. Explore equivalent RA plans
3. Find the RA plan with cheapest estimated cost
4. Convert RA to code and execute

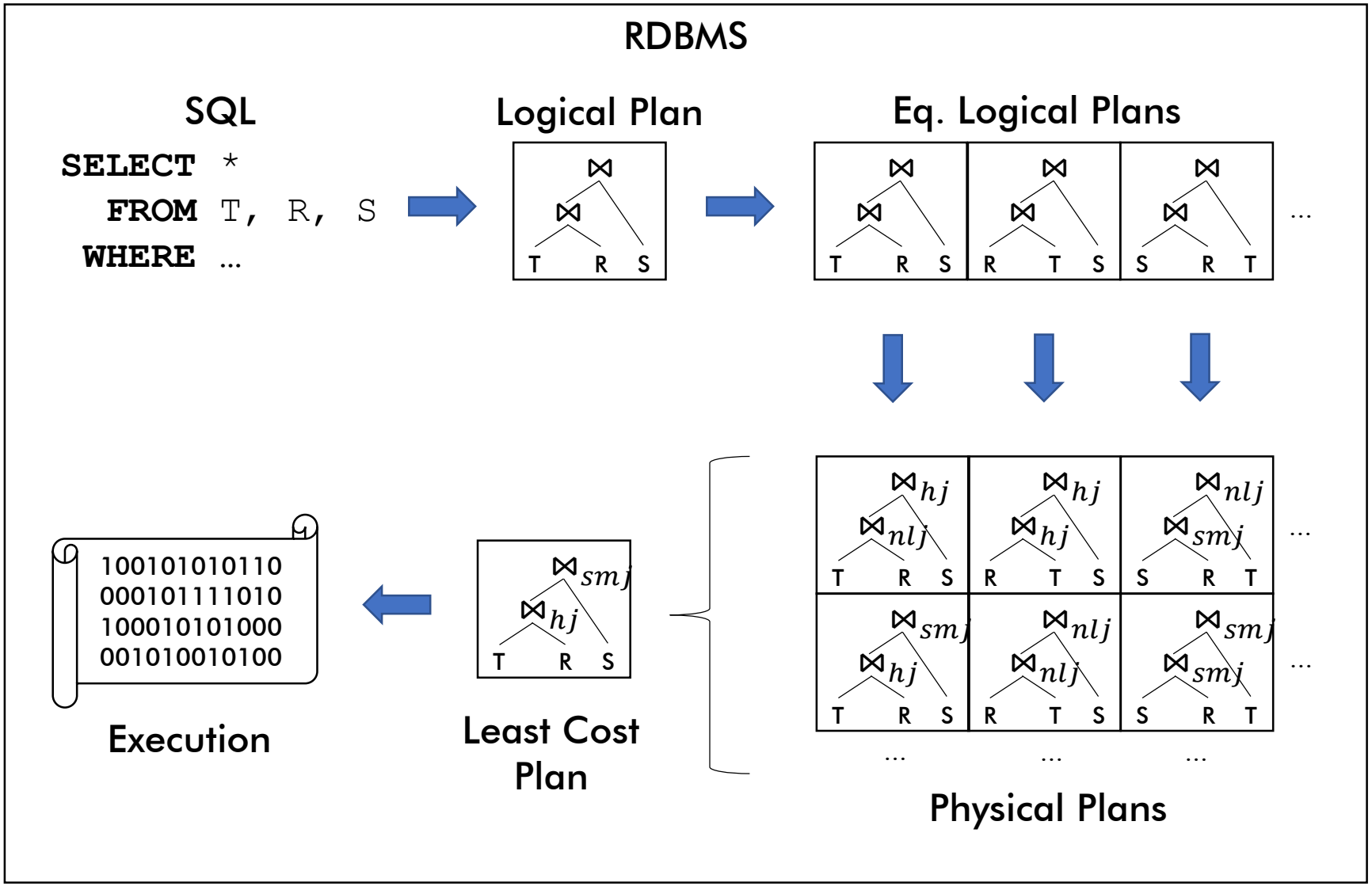
Today:
Cardinality Estimation



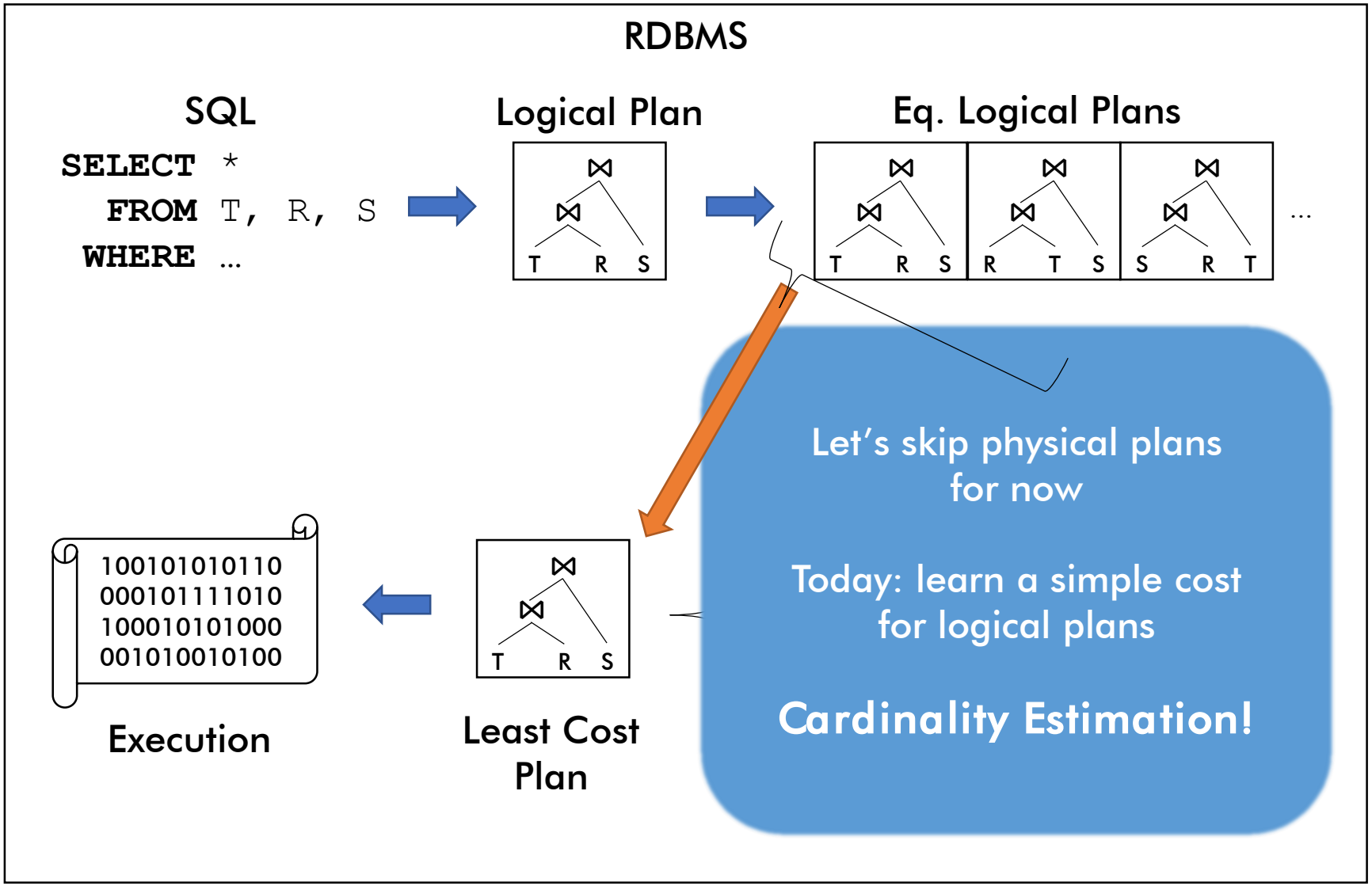
Query Optimization



Query Optimization



Query Optimization



Cardinality Estimation

- Estimate the number of tuples in the output of each RA operator
  - err, without actually computing the output

- Let's go grocery shop!
  - Safeway(id, name, category, price)
  - QFC(id, name, category, price)

- Let's use store stats to estimate the cardinality of some queries

Cardinality Estimation

Underline = primary key

- Safeway(id, name, category, price)
  - T = 1000 *# of tuples*
  - V(name) = 900 *# of distinct values*
  - V(category) = 10
  - V(price) = 200
  - Range(price) = [1,50) *range of values*
- QFC(id, name, category, price)
  - T = 2000
  - V(name) = 1900
  - V(category) = 12
  - V(price) = 500

Cardinality Estimation: SELECT

Safeway(<u>id</u>, name, category, price) T = 1000

```
SELECT name
  FROM Safeway
```

How many tuples do we expect this query to output?

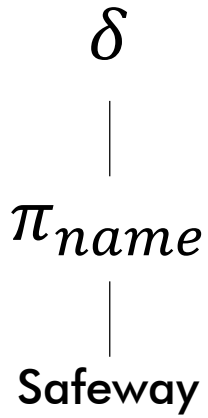
ANSWER: 1000 (no change)

Cardinality Estimation: DISTINCT

Safeway(id, name, category, price) T = 1000

V(name) = 900

```
SELECT DISTINCT name
  FROM Safeway
```



How many tuples do we expect this query to output?

ANSWER: 900 (set to distinct values)

Cardinality Estimation: AGGREGATE

Safeway(id, name, category, price) T = 1000

V(category) = 10

```
SELECT category,
       AVG(price)
  FROM Safeway
 GROUP BY category
```

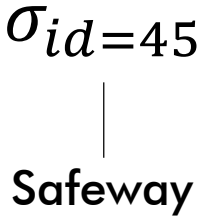
How many tuples do we expect this query to output?

ANSWER: 10 (set to distinct values)

Cardinality Estimation: WHERE Value

Safeway(id, name, category, price) T = 1000

```
SELECT *
  FROM Safeway
 WHERE id = 45
```



How many tuples do we expect this query to output?

ASSUME: that '45' exists in the distinct values of id

Answer is 0 otherwise…

ANSWER: 1

Cardinality Estimation: WHERE Value

Safeway(id, name, category, price) T = 1000

V(name) = 900

```
SELECT *
FROM Safeway
WHERE name = 'Milk'
```

$\sigma_{name="Milk"}$

|

Safeway

ASSUME: distinct values uniformly distributed

Without assumptions, estimation is impossible…

ANSWER: 1000 / 900 ≈ 1.11 tuples

Cardinality Estimation: WHERE Value

Safeway(id, name, category, price) T = 1000

V(name) = 900

```
SELECT *
FROM Safeway
WHERE name = 'Milk'
```

$\sigma_{name="Milk"}$

|

Safeway

Select Value: $\frac{T(op)}{V(op, attr)}$

ASSUME: distinct values uniformly distributed

Without assumptions, estimation is impossible…

ANSWER: 1000 / 900 ≈ 1.11 tuples

The selectivity factor

Cardinality Estimation: WHERE Range

Safeway(<u>id</u>, name, category, price) T = 1000

V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
```

$\sigma_{price<20}$

|

Safeway

ASSUME: distinct values uniformly distributed & continuous

Without assumptions, estimation is impossible…

ANSWER: 1000 * (20 – 1) / (50 – 1) ≈ 387.8 tuples

Cardinality Estimation: WHERE Range

Safeway(id, name, category, price) T = 1000

V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
```

$\sigma_{price<20}$

|

Safeway

Select Range: $T(op) * \frac{(Val-Min)}{(Max-Min)}$

ASSUME: distinct values uniformly distributed & continuous

Without assumptions, estimation is impossible…

ANSWER: 1000 * (20 – 1) / (50 – 1) ≈ 387.8 tuples

The selectivity factor

Cardinality Estimation: INTERSECTION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
   AND name = 'Milk'
```

$$\sigma_{price<20\ AND\ name="Milk"}$$

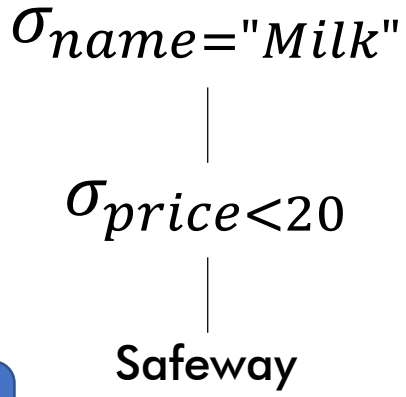
Safeway

Cardinality Estimation: INTERSECTION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
   AND name = 'Milk'
```



or use intersection

e.g. no milk costs < 20

e.g. all milk costs < 20

Hard to say

If conditions independent, **multiply** estimates

If conditions fully overlap, take **minimum** of estimates

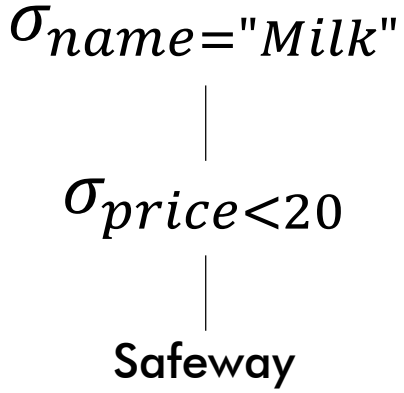
ASSUME independent unless you know for sure full overlap

Cardinality Estimation: INTERSECTION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
   AND name = 'Milk'
```



ANSWER:

≥ 1000 * [(20 – 1) / (50 – 1)] * 1/900 ≈ 0.431 tuples

≤ 1000 * min{(20 – 1) / (50 – 1), 1/900} ≈ 1.111 tuples

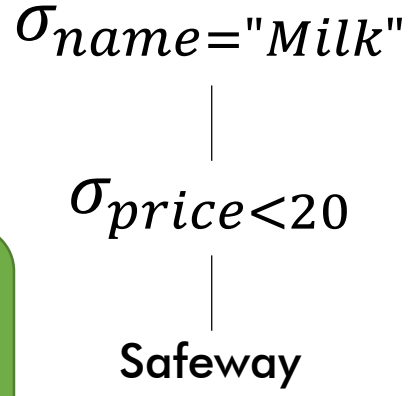
For this class, assume independence & answer 0.431 tuples

Cardinality Estimation: INTERSECTION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
```



AND / INTERSECT: $T(op) * cond1 * cond2$

unless full overlap: $T(op) * \min\{cond1, cond2\}$

The selectivity factor

≥ 1000 * [(20 – 1) / (50 – 1)] * 1/900 ≈ 0.431 tuples

≤ 1000 * min{(20 – 1) / (50 – 1), 1/900} ≈ 1.111 tuples

For this class, assume independence & answer 0.431 tuples

Cardinality Estimation: UNION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
    OR name = 'Milk'
```

$$\sigma_{price<20\ OR\ name="Milk"}$$

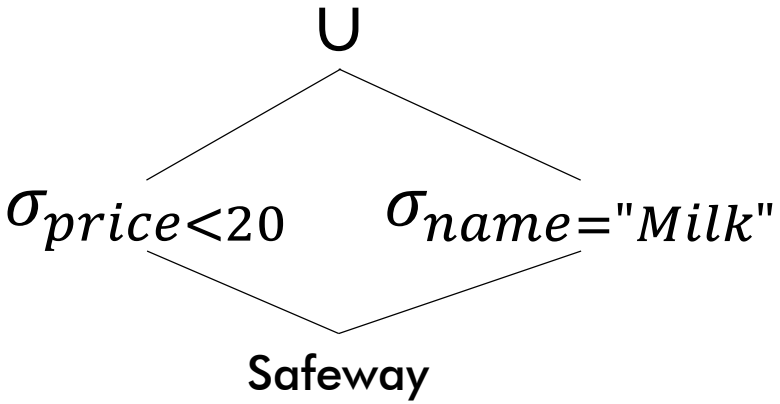
Safeway

Cardinality Estimation: UNION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
    OR name = 'Milk'
```



Hard to say

If conditions independent, **add** estimates

If conditions fully overlap, take **minimum** of estimates

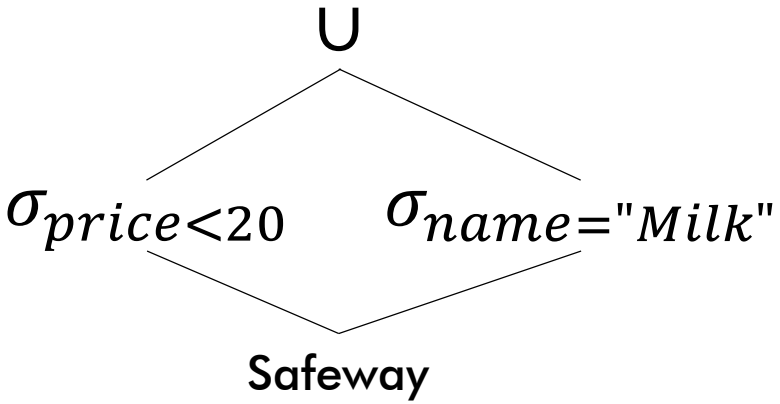
ASSUME independent unless you know for sure full overlap

Cardinality Estimation: UNION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
    OR name = 'Milk'
```



ANSWER:

≥ 1000 * min{(20 – 1) / (50 – 1), 1/900} ≈ 1.111 tuples

≤ 1000 * (20 – 1) / (50 – 1) + 1000 / 900 ≈ 388.9 tuples

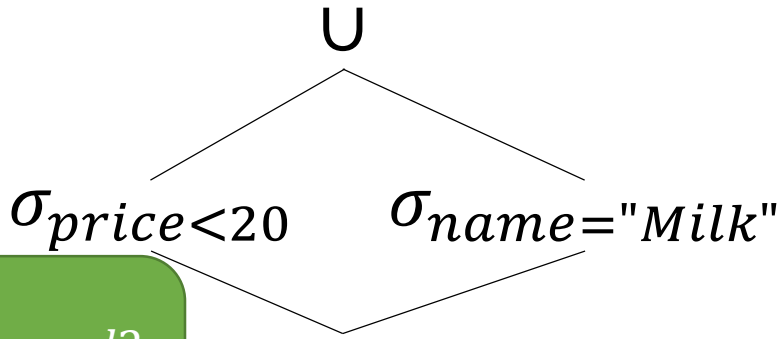
For this class, assume independence & answer 388.9 tuples

Cardinality Estimation: UNION

Safeway(id, name, category, price) T = 1000

V(name) = 900 V(price) = 200 Range(price) = [1,50)

```
SELECT *
FROM Safeway
WHERE price < 20
```



OR / UNION: $T(op) * cond1 + T(op) * cond2$

unless full overlap: $T(op) * \min\{cond1, cond2\}$

The selectivity factor

Al

≥ 1000 * min{(20 – 1) / (50 – 1), 1/900} ≈ 1.111 tuples

≤ 1000 * (20 – 1) / (50 – 1) + 1000 / 900 ≈ 388.9 tuples

For this class, assume independence & answer 388.9 tuples

Cardinality Estimation: CARTESIAN PROD

Safeway(id, name, category, price)

T = 1000 V(name) = 900 V(category) = 10

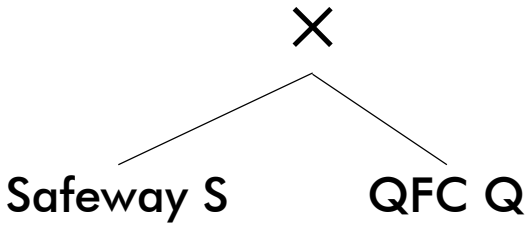
V(price) = 200 Range(price) = [1,50)

QFC(id, name, category, price)

T = 2000 V(name) = 1900 V(category) = 12

V(price) = 500

```
SELECT *
FROM Safeway S, QFC Q
WHERE TRUE
```



No selectivity factor

ANSWER: 1000 * 2000

= 2000000 tuples

Cartesian Product: $T(op1) * T(op2)$

Cardinality Estimation: JOIN

Safeway(id, name, category, price)

T = 1000 V(name) = 900 V(category) = 10

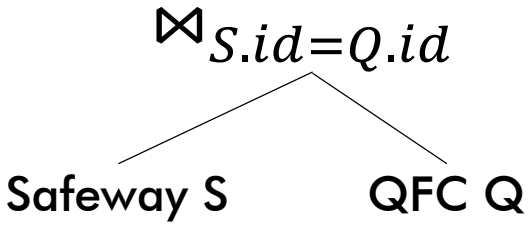
V(price) = 200 Range(price) = [1,50)

QFC(id, name, category, price)

T = 2000 V(name) = 1900 V(category) = 12

V(price) = 500

```
SELECT *
FROM Safeway S, QFC Q
WHERE S.id = Q.id
```



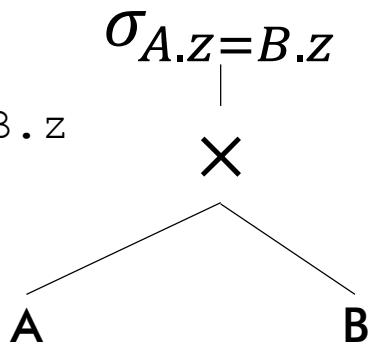
ANSWER: ≤ 1000 * 2000

Can we do better? Let's look at joins in general first…

Cardinality Estimation: JOIN

1. T(A) * T(B) tuples in Cartesian product
2. Suppose z0 exists in the join
3. How many times does z0 occur?
   - Like the selection condition $\sigma_{A.z=z0\ AND\ B.z=z0}$
4. How many distinct z0s exist in the join?
   - ≥ 0 [if no overlap]
   - ≤ min{V(A, z), V(B, z)} [if full overlap]
   - For this class, ASSUME full overlap
5. Multiply by estimate # of distinct z0s

```
SELECT *
FROM A, B
WHERE A.z = B.z
```



Selectivity Factor
1/V(A, z) * 1/V(B, z)

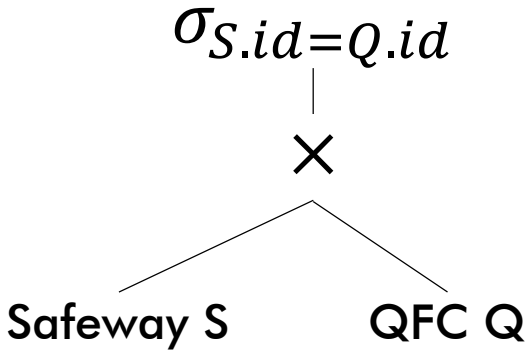
$$\frac{T(A) * T(B)}{V(A,z) * V(B,z)} * \min\{V(A,z), V(B,z)\} = \frac{T(A) * T(B)}{\max\{V(A,z),\ V(B,z)\}}$$

Cardinality Estimation: JOIN

Safeway(id, name, category, price) -- T = 1000, V(id) = 1000

QFC(id, name, category, price) -- T = 2000, V(id) = 2000

```
SELECT *
FROM Safeway S, QFC Q
WHERE S.id = Q.id
```



$$\frac{T(Safeway) * T(QFC)}{\max\{V(Safeway, id),\ V(QFC, id)\}} = \frac{1000 * 2000}{\max\{1000, 2000\}} = 1000 \text{ tuples}$$

Right… if we assume full overlap of ids between Safeway & QFC…

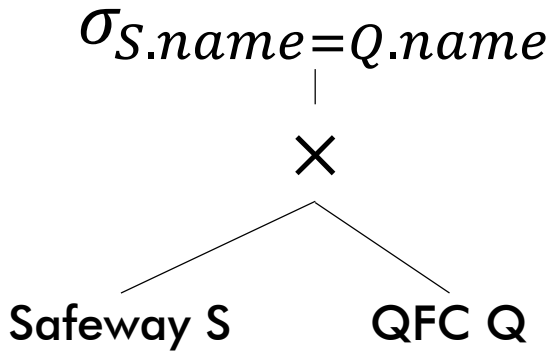
then we only need keep the ids of the smaller (Safeway)

Cardinality Estimation: JOIN

Safeway(id, name, category, price) -- T = 1000, V(name) = 900

QFC(id, name, category, price) -- T = 2000, V(name) = 1900

```
SELECT *
FROM Safeway S, QFC Q
WHERE S.name = Q.name
```



$$\frac{T(Safeway) * T(QFC)}{\max\{V(Safeway, name),\ V(QFC, name)\}} = \frac{1000 * 2000}{\max\{900, 1900\}} \approx 1052.6 \text{ tuples}$$

Right… if there are less distinct values (more repeats)…

then the join results in more tuples

Cardinality for Optimization

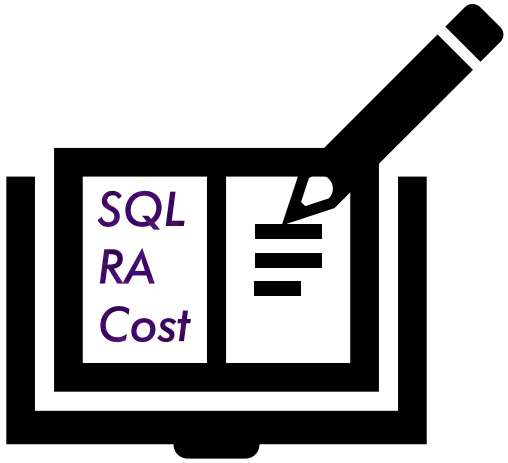
- Now we know how to estimate RA cardinality
- Optimization: find the equivalent RA Plan that **minimizes operator cardinality**
  - Often we only care about the operator with largest cardinality (most "expensive")
  - "MinMax"

- Real RDBMS uses sophisticated cost models
  - I/O estimate in reads/writes
  - Compute estimate in FLOPS
  - Memory estimate in bytes

Practice!

- Find practice SQL problems online
- Draw their RA
- Rearrange the RA (w/ equivalence rules)

- Make up input statistics
- Estimate cardinalities
- Find RA with cheaper cardinality



- Practice with a partner! Post Piazza! Ask TAs!