CSE 344 Introduction to Data Management

Section 4: Relational Algebra

Outline

- HW3 Check-in
- Relational Algebra Review
- Translate nested SQL Queries to RA
- Translate from RA to SQL

Relational Algebra

- SQL = WHAT we want to get from the data
- Relational Algebra = HOW to get the data we want
- SQL → Relational Algebra →Physical Plan
- Relational Algebra = Logical Plan (usually written as a tree)

Relational Algebra Operators

Standard:

- Selection: σ
  Projection: π
  Rename: ρ

Sets:

- Union: ∪
  Intersection: ∩
  Difference: -

Joins:

- Cartesian Product: X
  Join: ⋈

Extended:

- Duplicate Elimination: δ
  Grouping and Aggregation: ɣ
  Sorting: τ

SQL to RA Review

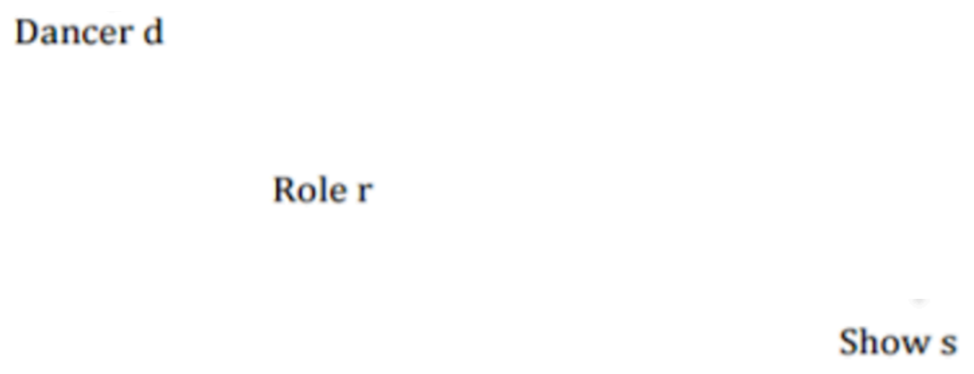
Dancer(did, name, birthyear, country)
Show(sid, title, choreographer, composer, year)
Role(did, sid, role, company)

- Write a Relational Algebra plan for the following query:

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```

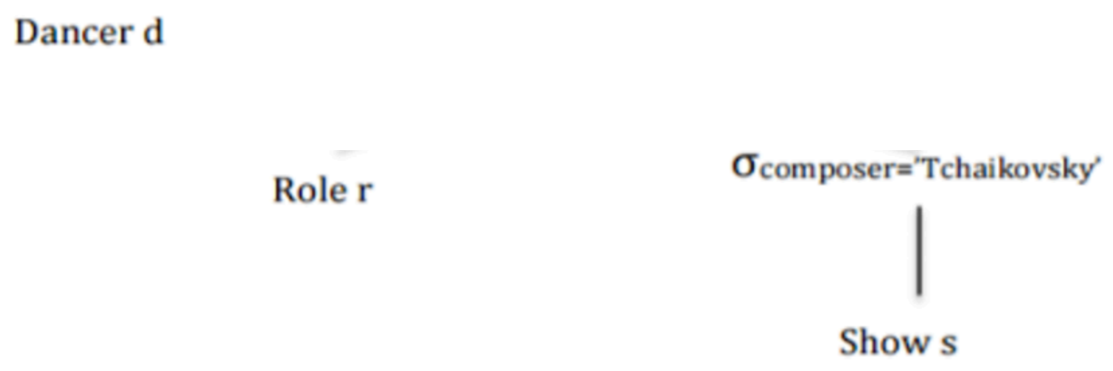
SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```



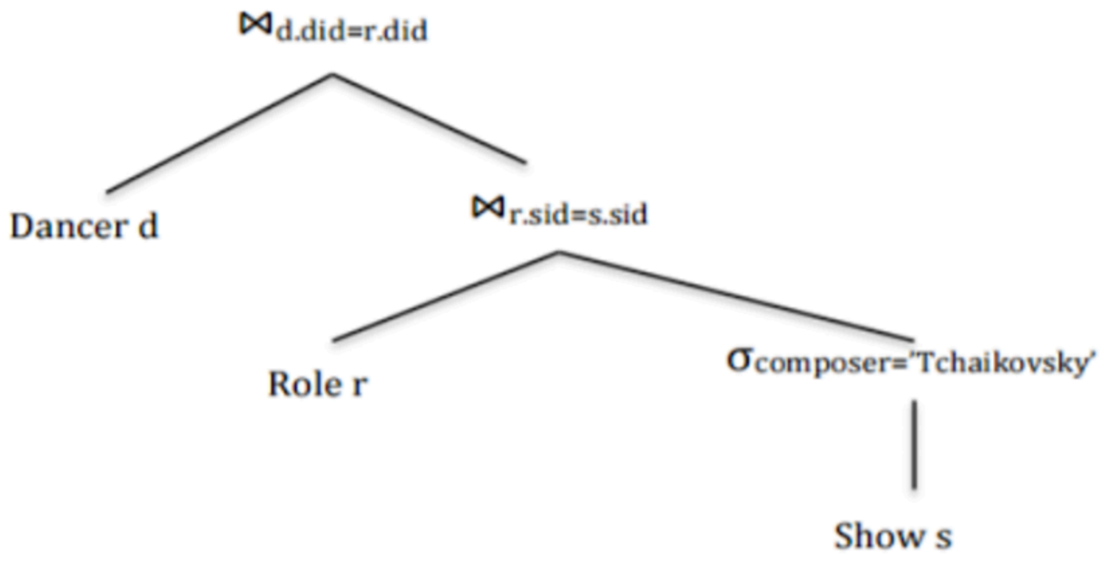
SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```



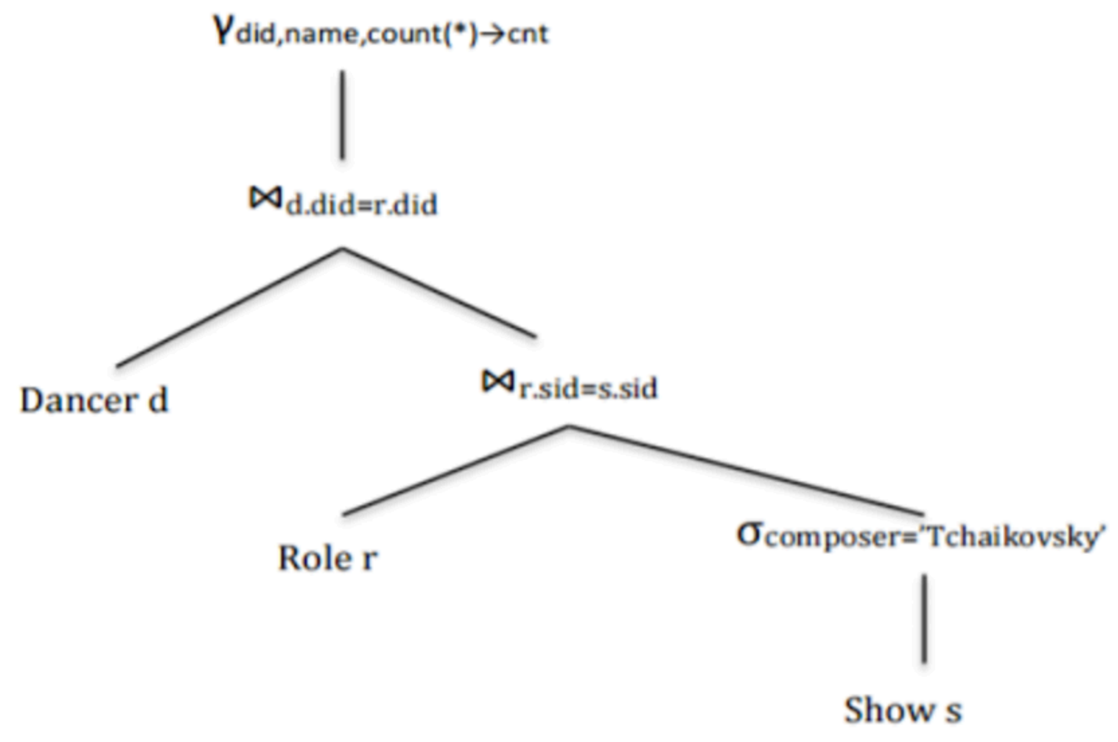
SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```



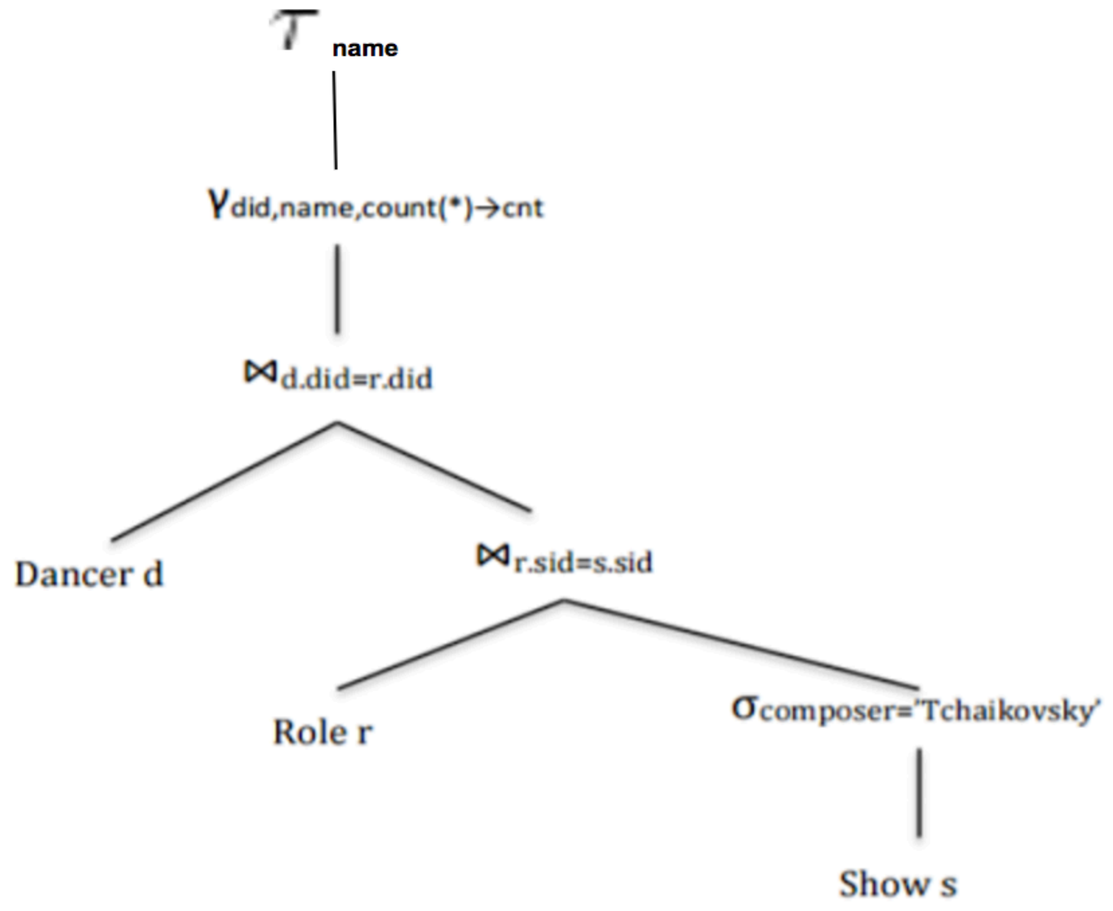
SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```


$\gamma_{did,name,count(*)\rightarrow cnt}$
$\bowtie_{d.did=r.did}$
Dancer d
$\bowtie_{r.sid=s.sid}$
Role r
$\sigma_{composer='Tchaikovsky'}$
Show s

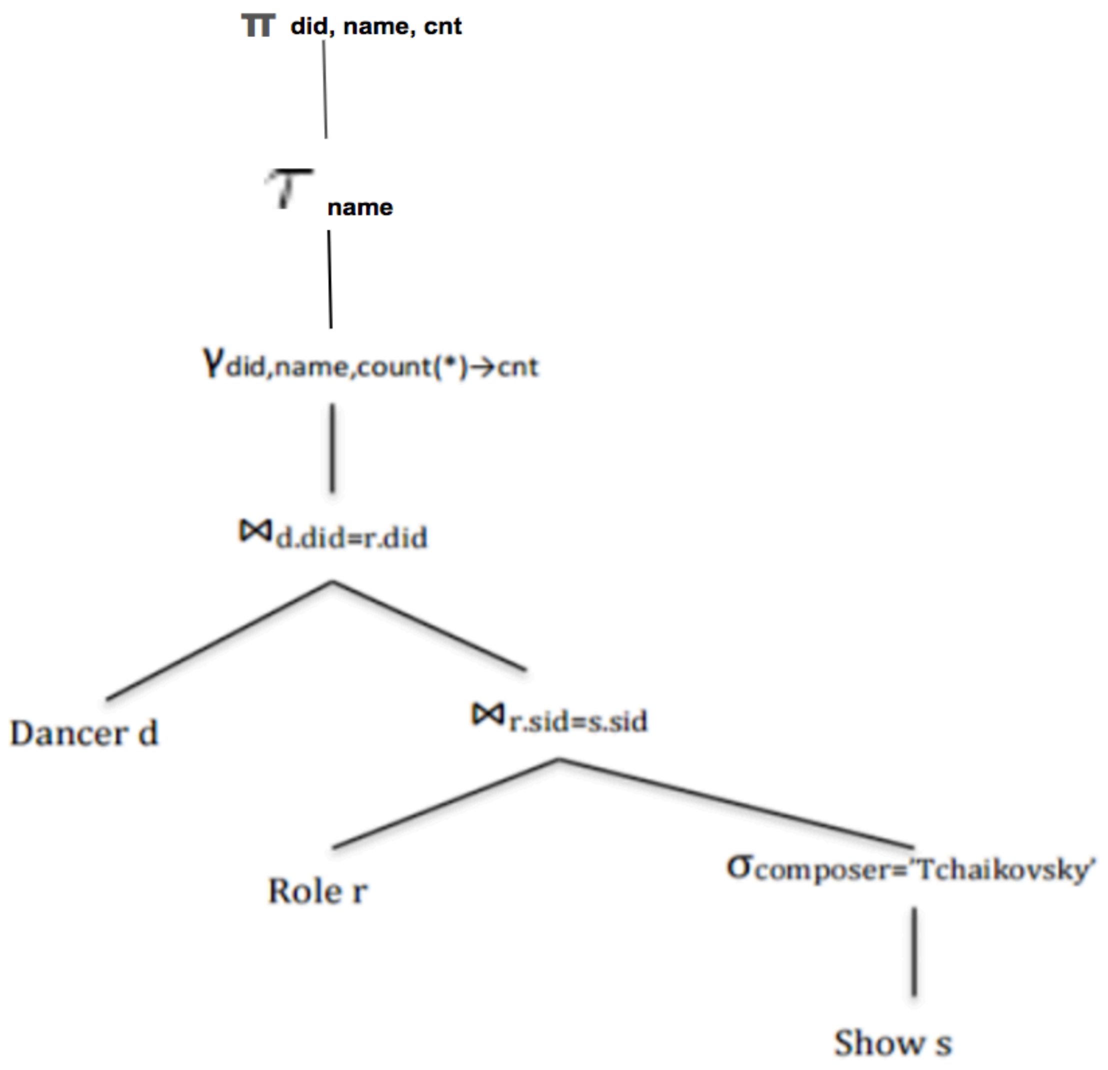
SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```



SQL to RA Solution

```
SELECT d.did, d.name, count(*)
FROM Dancer d, Show s, Role r
WHERE d.did=r.did
AND r.sid=s.sid
AND s.composer='Tchaikovsky'
GROUP BY d.did, d.name
ORDER BY d.name;
```



Translate nested SQL Queries to RA

Member(mid, name, age)

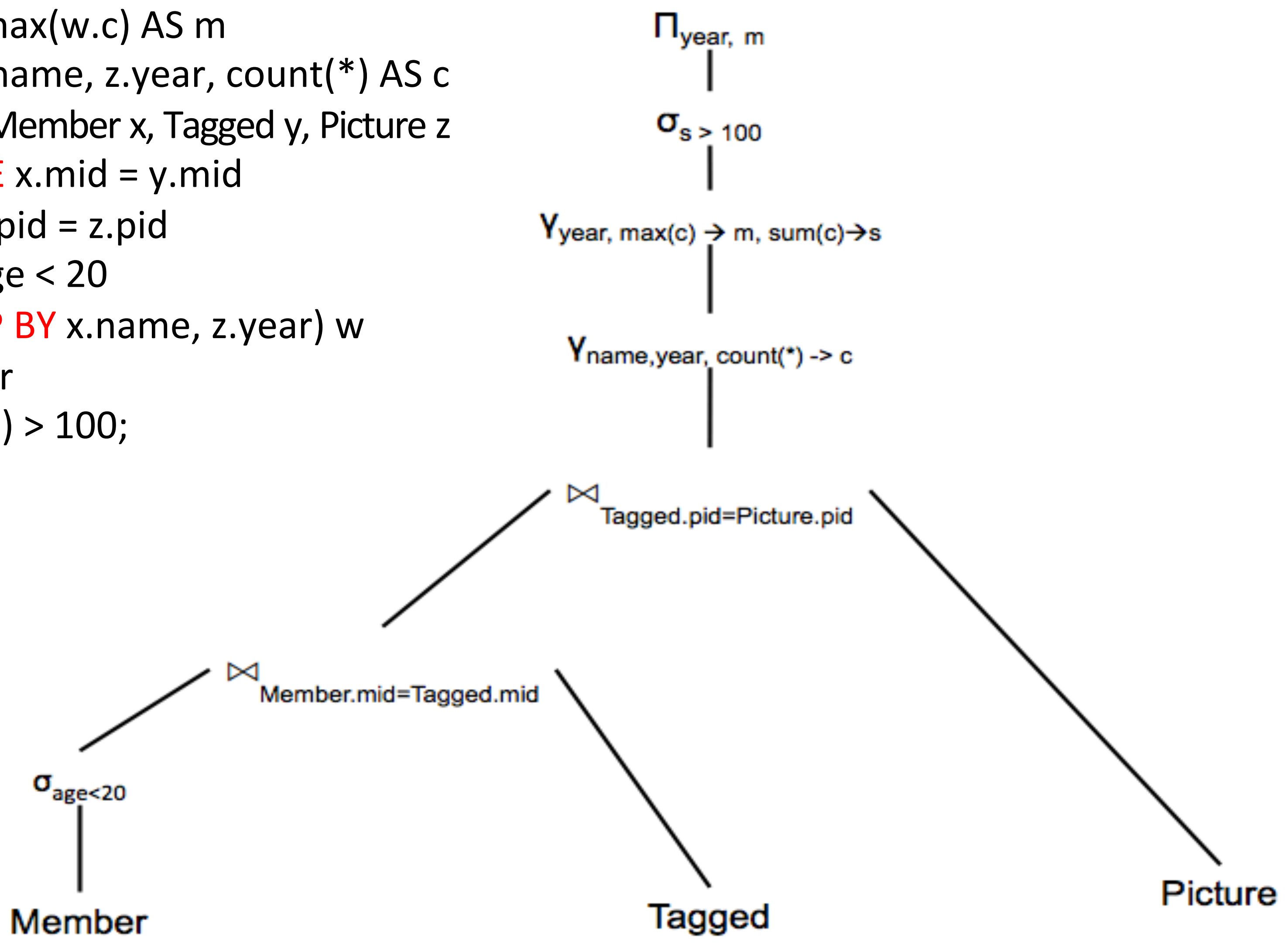
Picture(pid, year)

Tagged(mid, pid)

```
SELECT w.year, max(w.c) AS m
FROM(SELECT x.name, z.year, count(*) AS c
         FROMMember x, Tagged y, Picture z
         WHERE x.mid = y.mid
         AND y.pid = z.pid
         AND age < 20
         GROUP BY x.name, z.year) w
GROUP BY w.year
HAVING sum(w.c) > 100;
```

Nested SQL Queries to RA Solution

```
SELECT w.year, max(w.c) AS m
FROM(SELECT x.name, z.year, count(*) AS c
          FROM Member x, Tagged y, Picture z
          WHERE x.mid = y.mid
          AND y.pid = z.pid
          AND age < 20
          GROUP BY x.name, z.year) w
GROUP BY w.year
HAVING sum(w.c) > 100;
```

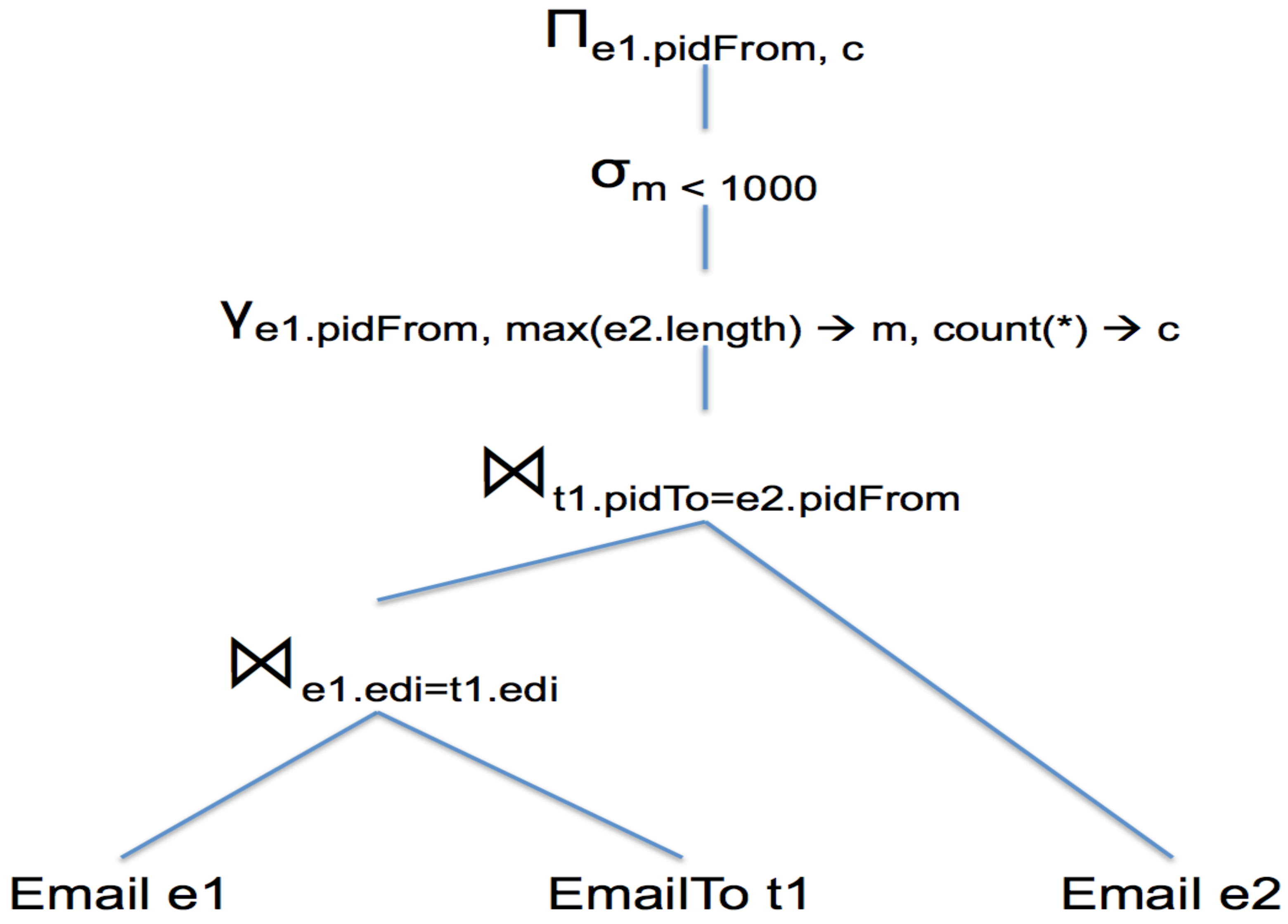

$\Pi_{year,\ m}$
$\sigma_{s > 100}$
$\gamma_{year,\ max(c) \rightarrow m,\ sum(c) \rightarrow s}$
$\gamma_{name,year,\ count(*) \rightarrow c}$
$\bowtie_{Tagged.pid=Picture.pid}$
$\bowtie_{Member.mid=Tagged.mid}$
$\sigma_{age<20}$
Member
Tagged
Picture

Translate from RA to SQL

- Put tables in FROM clause
- Put join predicates in WHERE clause
- Put selection predicates in WHERE clause
- Translate extended RA symbols to SQL equivalent
- Put selection of aggregates in HAVING clause
- Put projection predicates in SELECT clause

RA to SQL Example

Person(pid,name)
Email(eid, pidFrom, tid, body, length)
EmailTo(eid,pidTo)


$\Pi_{e1.pidFrom,\ c}$
$\sigma_{m < 1000}$
$\gamma_{e1.pidFrom,\ max(e2.length) \rightarrow m,\ count(*) \rightarrow c}$
$\bowtie_{t1.pidTo=e2.pidFrom}$
$\bowtie_{e1.edi=t1.edi}$
Email e1
EmailTo t1
Email e2

RA to SQL Solution

```
SELECT e1.pidFrom, count(*)
FROM Email e1, EmailTo t1, Email e2
WHERE e1.eid = t1.eid
AND t1.pidTo = e2.pidFrom
GROUP BY e1.pidFrom
HAVING max(e2.length) < 1000;
```