Part 1

Monday, January 17, 2022 8:20 PM

Without Joins

```
WITH directors AS
    (SELECT id, pid FROM crew
     WHERE crewtype = 'director'),
 main_roles AS
     (SELECT id, pid FROM roles
      WHERE roletype = 'actor' OR roletype = 'actress')
 SELECT count(*) FROM main_roles AS mr
 WHERE EXISTS
     (SELECT id, pid FROM directors AS d
        WHERE d.pid = mr.pid AND d.id = mr.id
    );
```

Relational Algebra

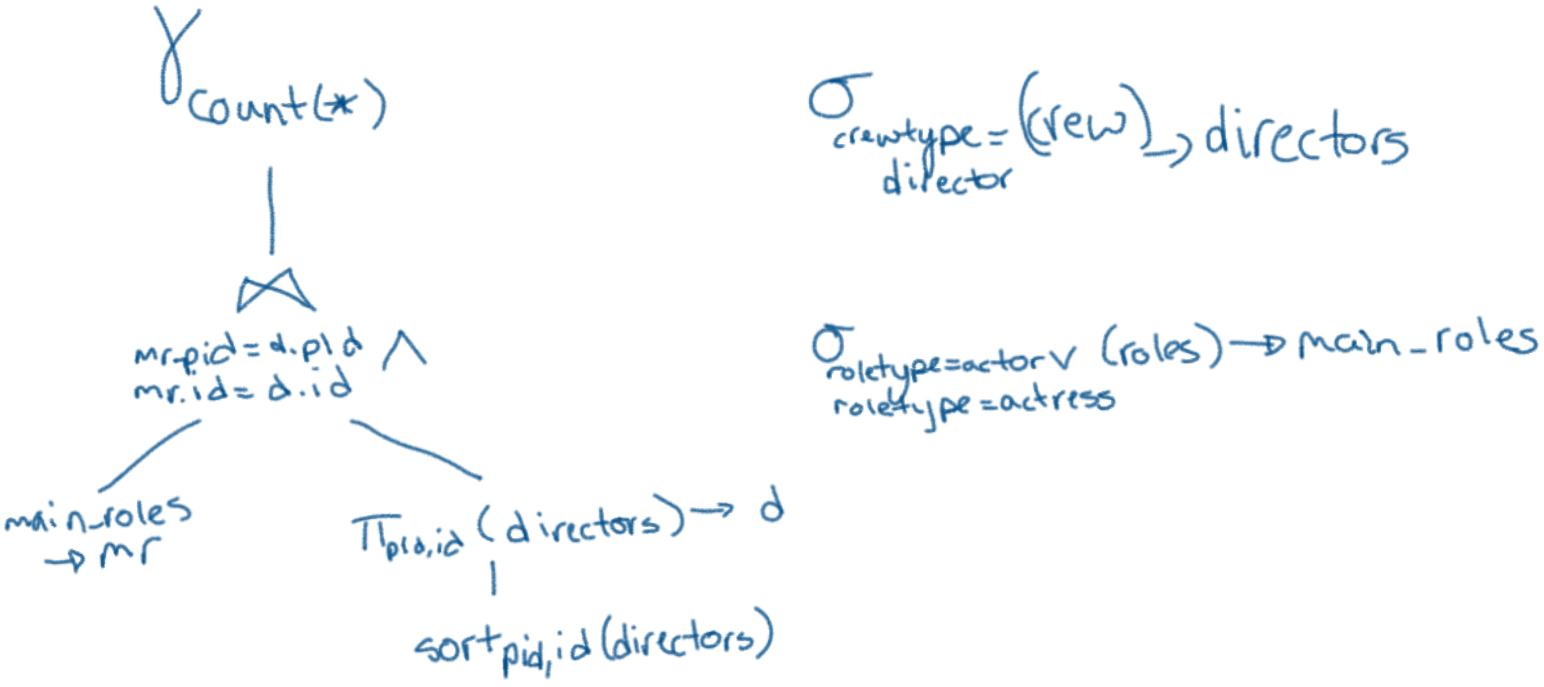
$$directors = \pi_{id,pid}(\sigma_{crewtype=director}(crew))$$

$$main\_roles = \pi_{id,pid}\left(\sigma_{roletype=actor \vee roletype=actress}(roles)\right)$$

$$\gamma_{count(*)}(\sigma_{Exists\left(\sigma_{d.pid=mr.pid \wedge d.id=mr.id}(directors \to d)\right)}(main\_roles \to mr))$$

Explain

```
Aggregate  (cost=2726921.02..2726921.03 rows=1 width=8)
  CTE directors
    ->  Seq Scan on crew  (cost=0.00..370255.20 rows=4846557 width=20)
          Filter: (crewtype = 'director'::text)
  CTE main_roles
    ->  Seq Scan on roles  (cost=0.00..595519.40 rows=14354682 width=20)
          Filter: ((roletype = 'actor'::text) OR (roletype = 'actress'::text))
  ->  Hash Join  (cost=1036925.82..1752174.75 rows=3588670 width=0)
        Hash Cond: ((mr.pid = d.pid) AND (mr.id = d.id))
        ->  CTE Scan on main_roles mr  (cost=0.00..287093.64 rows=14354682 width=64)
        ->  Hash  (cost=1035895.82..1035895.82 rows=40000 width=64)
              ->  Unique  (cost=999546.64..1035895.82 rows=40000 width=64)
                    ->  Sort  (cost=999546.64..1011663.04 rows=4846557 width=64)
                          Sort Key: d.pid, d.id
                          ->  CTE Scan on directors d  (cost=0.00..96931.14 rows=4846557 width=64)
```



With Joins

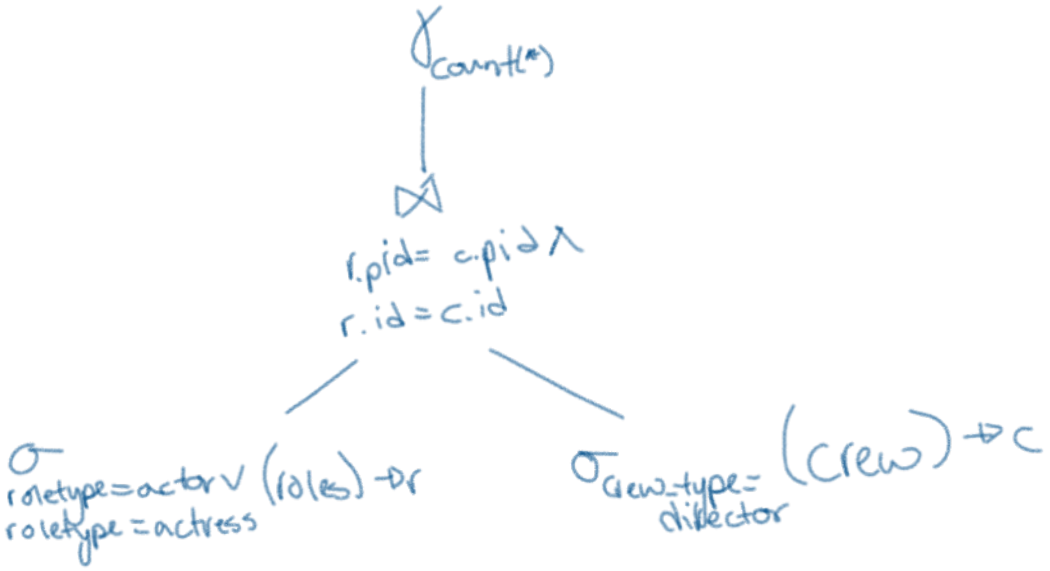
```
SELECT COUNT(*) FROM crew AS c
    JOIN roles AS r ON c.pid = r.pid AND c.id = r.id
    WHERE c.crewtype = 'director'
    AND (r.roletype = 'actor' OR r.roletype = 'actress');
```

Relational Algebra

$$\gamma_{count(*)}(\sigma_{c.crewtype=director \wedge (r.roletype=actress \vee r.roletype=actor)}(crew \rightarrow c \bowtie_{c.pid=r.pid \wedge c.id=r.id} roles \rightarrow r)$$

Explain

```
Finalize Aggregate  (cost=977012.99..977013.00 rows=1 width=8)
  -> Gather  (cost=977012.77..977012.98 rows=2 width=8)
       Workers Planned: 2
       -> Partial Aggregate  (cost=976012.77..976012.78 rows=1 width=8)
            -> Hash Join  (cost=471351.55..976011.92 rows=340 width=0)
                 Hash Cond: ((r.pid = c.pid) AND (r.id = c.id))
                 -> Parallel Seq Scan on roles r  (cost=0.00..374769.50 rows=5981118 width=20)
                      Filter: ((roletype = 'actor'::text) OR (roletype = 'actress'::text))
                 -> Hash  (cost=370255.20..370255.20 rows=4846557 width=20)
                      -> Seq Scan on crew c  (cost=0.00..370255.20 rows=4846557 width=20)
                           Filter: (crewtype = 'director'::text)
```



4. I would use the query that includes the JOIN. Although the DBMS realizes that the query can be better performed as a JOIN instead of constantly re-computing the "EXISTS" selection to see if the relation exists, it had the overhead of first sorting the subquery, then creating a hash on top of it, which resulted in a higher overall cost.