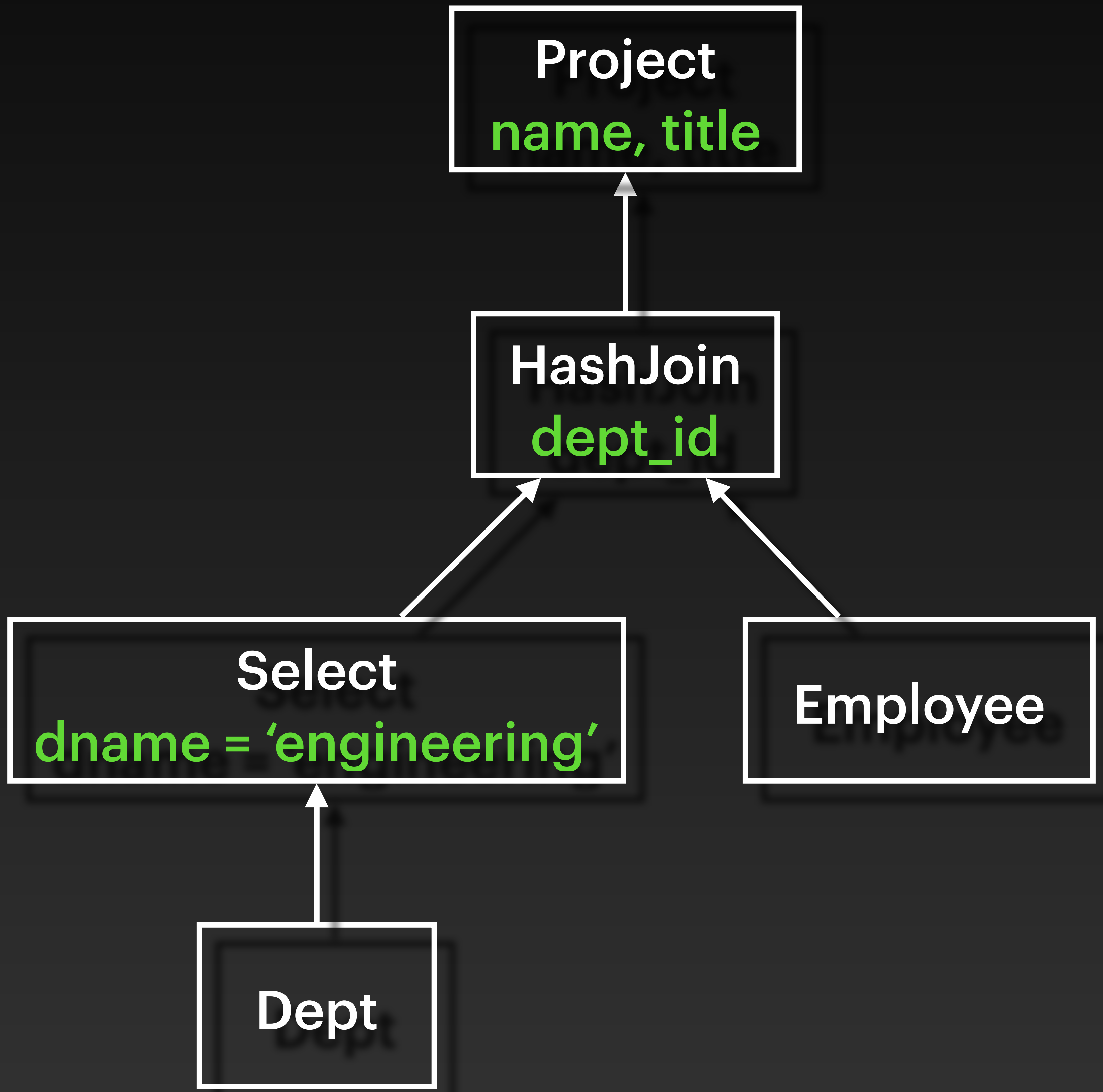
TECH VAULT

UNDERSTANDING “EXPLAIN” IN POSTGRES

AMR ELHELW

$$\pi_{\text{name,title}}(\text{Employee} \bowtie_{\text{dept\_id}} \sigma_{\text{dname='engineering'}}(\text{Dept}))$$



Project
name, title

HashJoin
dept_id

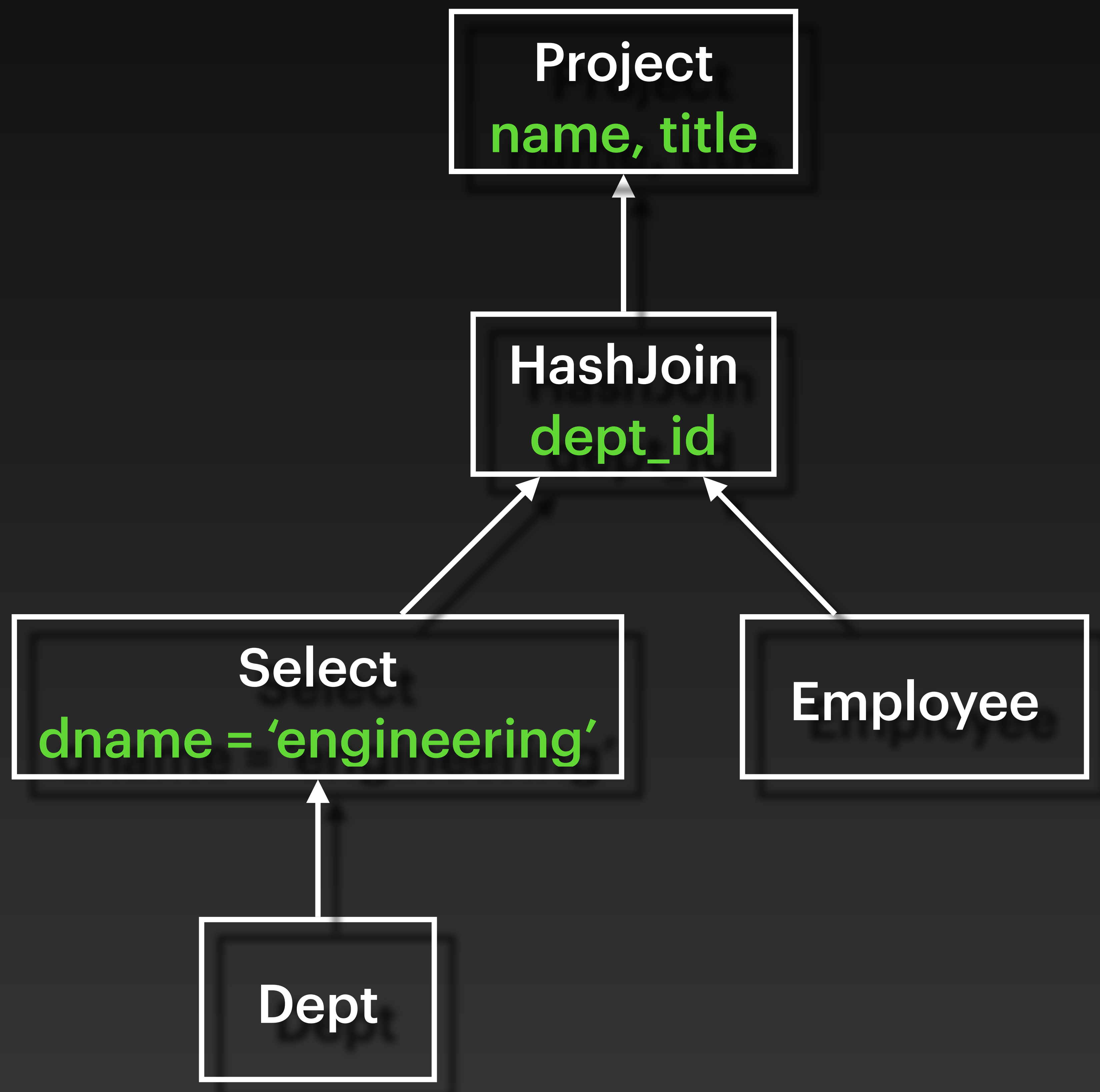
Select
dname = ‘engineering’

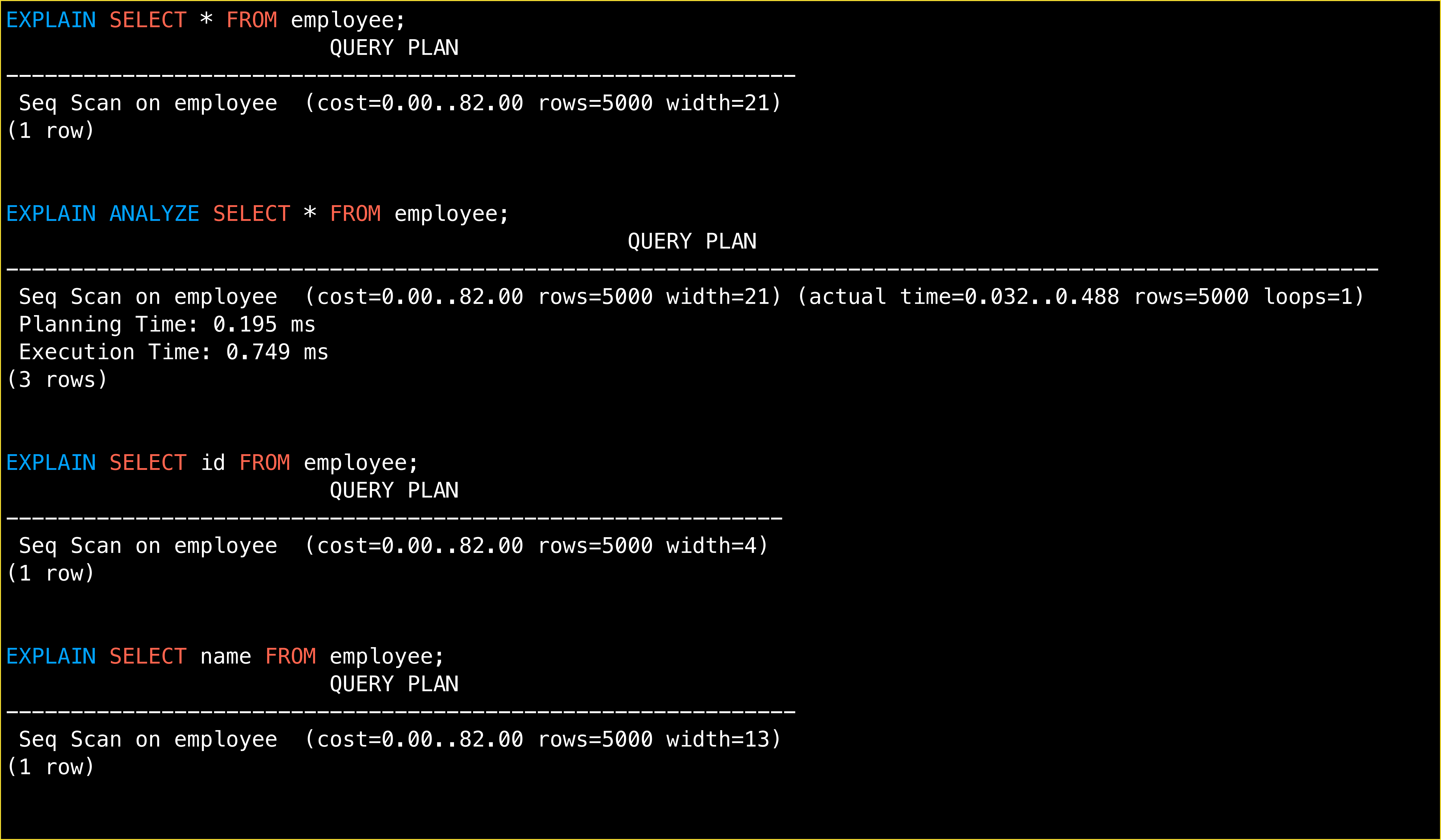
Employee

Dept

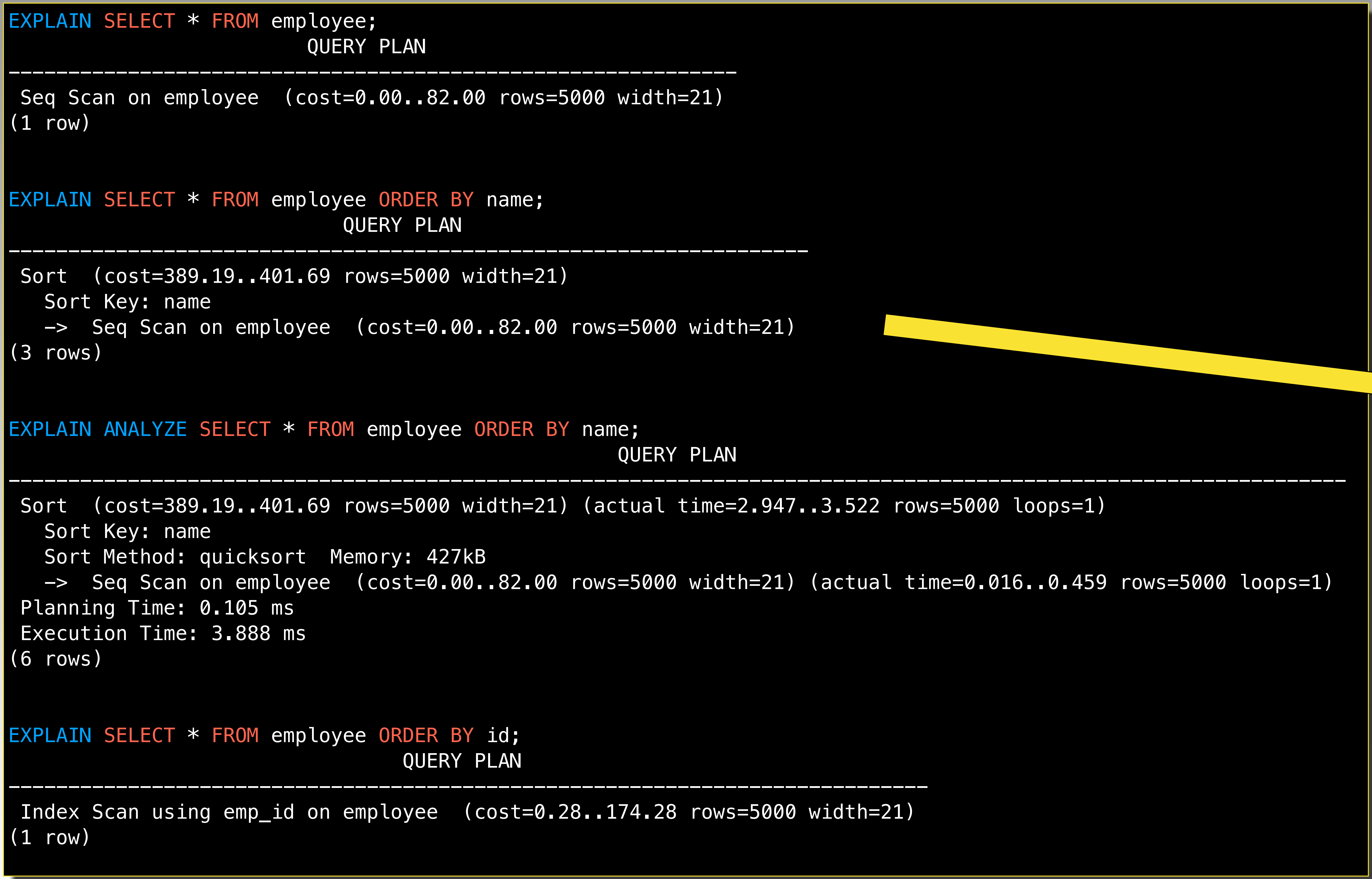
Node Types

- Leaf (no inputs)
- Unary (1 input)
- Binary (2 inputs)
- N-ary (*N* inputs, N > 2)

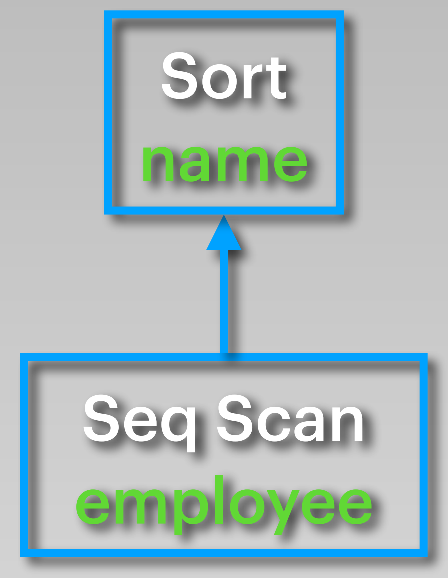




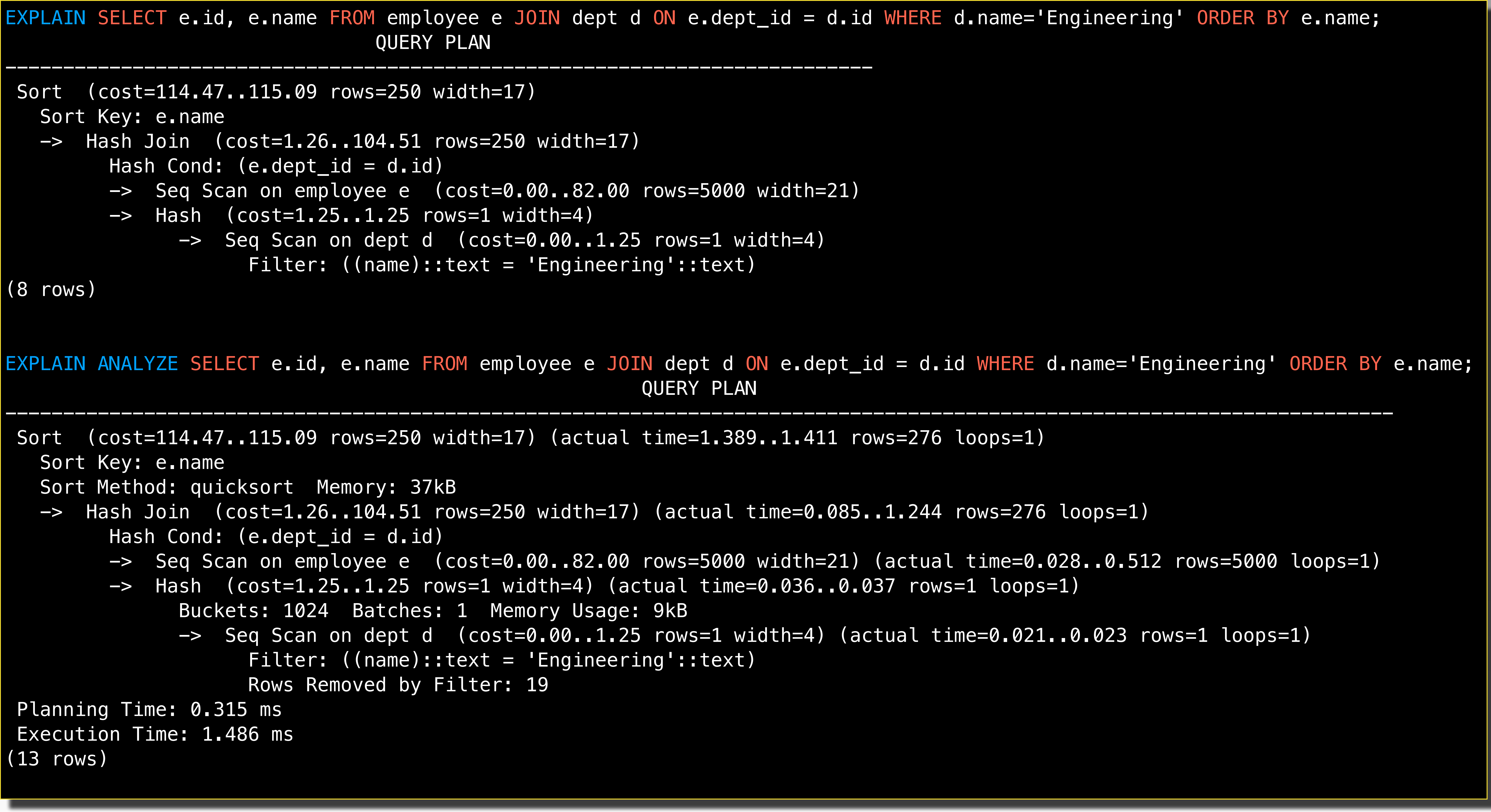
```
EXPLAIN SELECT * FROM employee;
                         QUERY PLAN
-------------------------------------------------------------
 Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=21)
(1 row)


EXPLAIN ANALYZE SELECT * FROM employee;
                                                QUERY PLAN
----------------------------------------------------------------------------------------------------------
 Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=21) (actual time=0.032..0.488 rows=5000 loops=1)
 Planning Time: 0.195 ms
 Execution Time: 0.749 ms
(3 rows)


EXPLAIN SELECT id FROM employee;
                         QUERY PLAN
------------------------------------------------------------
 Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=4)
(1 row)


EXPLAIN SELECT name FROM employee;
                         QUERY PLAN
-------------------------------------------------------------
 Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=13)
(1 row)
```



```
EXPLAIN SELECT * FROM employee;
                         QUERY PLAN
-------------------------------------------------------------
 Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=21)
(1 row)


EXPLAIN SELECT * FROM employee ORDER BY name;
                            QUERY PLAN
-------------------------------------------------------------------
 Sort  (cost=389.19..401.69 rows=5000 width=21)
   Sort Key: name
   ->  Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=21)
(3 rows)


EXPLAIN ANALYZE SELECT * FROM employee ORDER BY name;
                                                   QUERY PLAN
----------------------------------------------------------------------------------------------------------------
 Sort  (cost=389.19..401.69 rows=5000 width=21) (actual time=2.947..3.522 rows=5000 loops=1)
   Sort Key: name
   Sort Method: quicksort  Memory: 427kB
   ->  Seq Scan on employee  (cost=0.00..82.00 rows=5000 width=21) (actual time=0.016..0.459 rows=5000 loops=1)
 Planning Time: 0.105 ms
 Execution Time: 3.888 ms
(6 rows)


EXPLAIN SELECT * FROM employee ORDER BY id;
                                 QUERY PLAN
-----------------------------------------------------------------------------
 Index Scan using emp_id on employee  (cost=0.28..174.28 rows=5000 width=21)
(1 row)
```

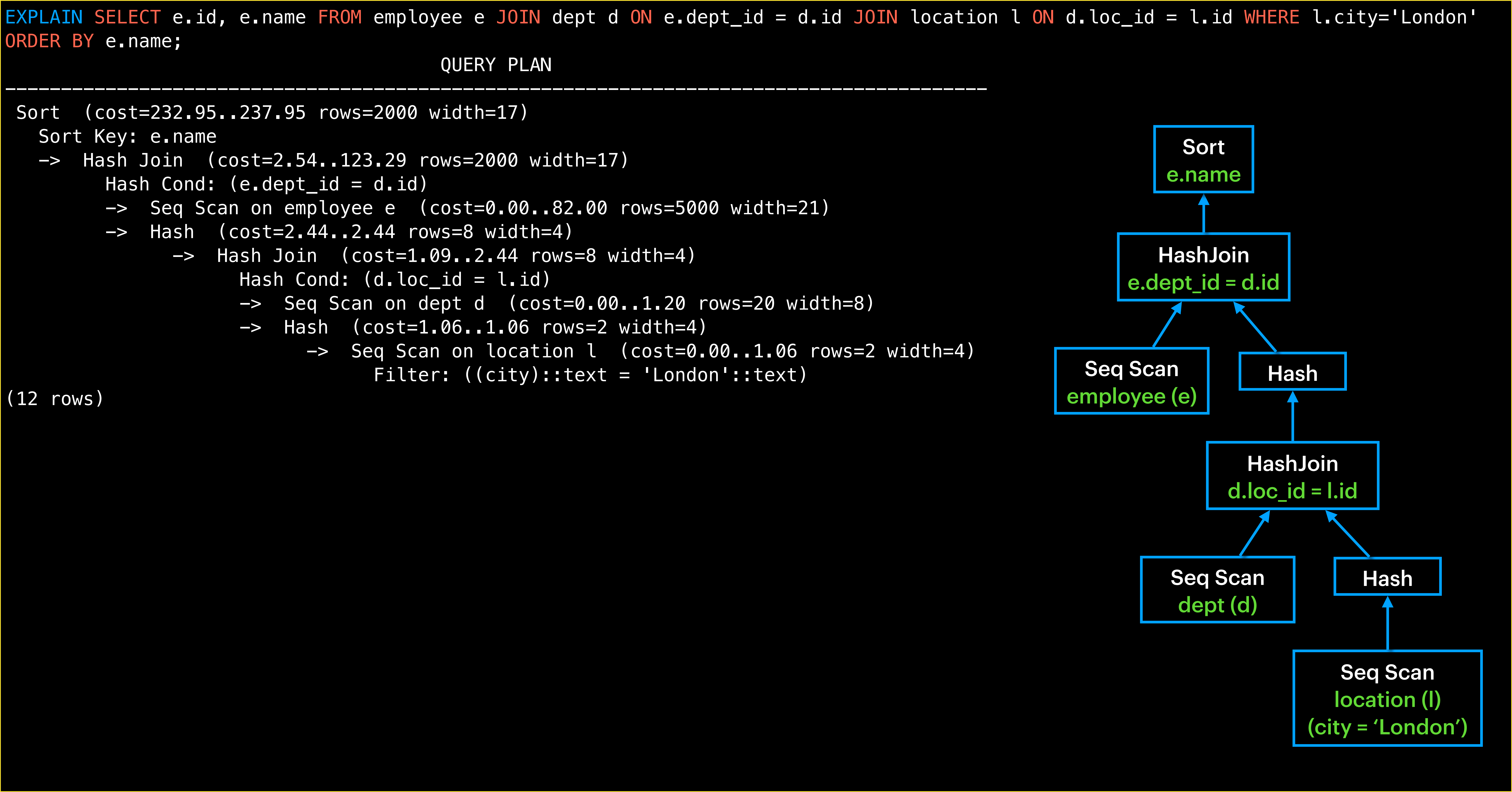


Amr Elhelw's
TECH
VAULT



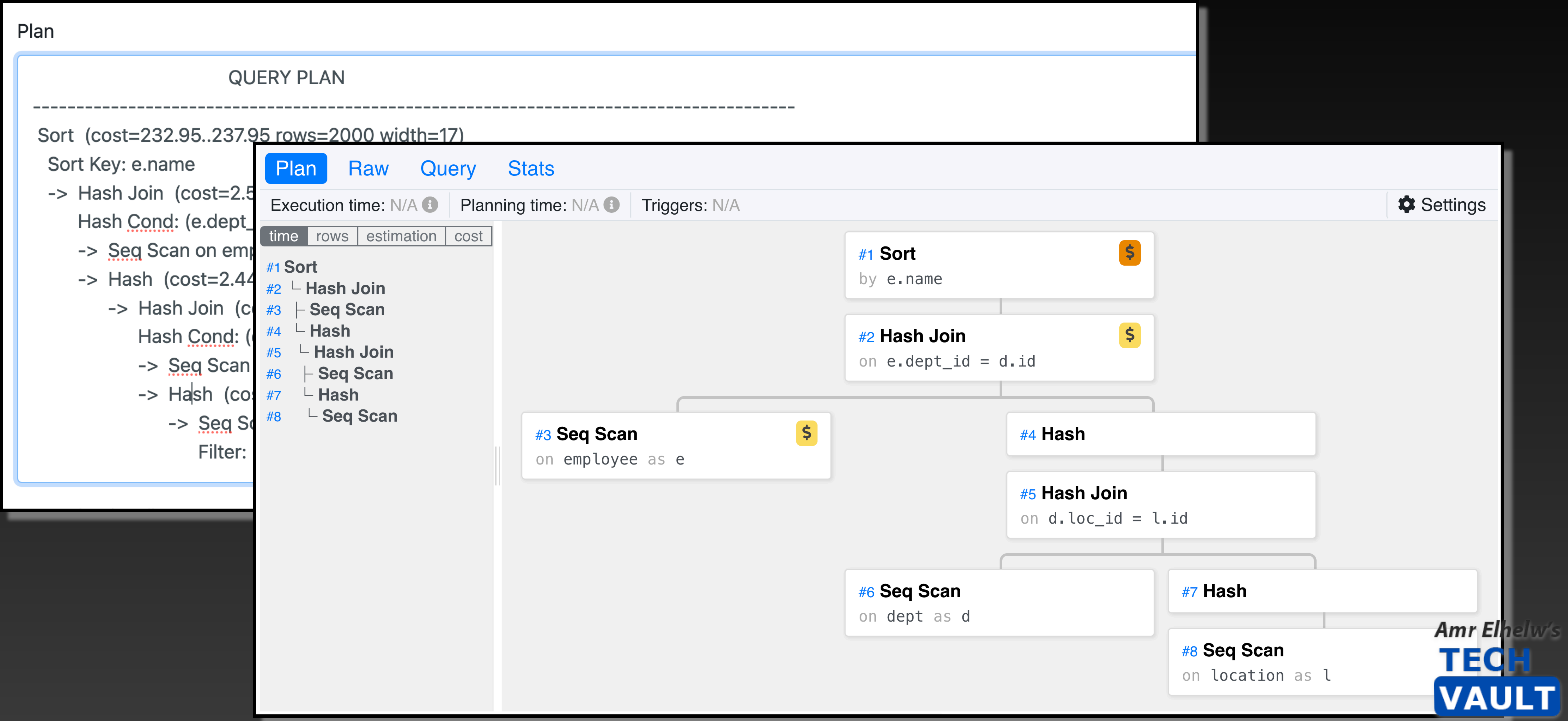
```
EXPLAIN SELECT e.id, e.name FROM employee e JOIN dept d ON e.dept_id = d.id WHERE d.name='Engineering' ORDER BY e.name;
                                QUERY PLAN
---------------------------------------------------------------------------
 Sort  (cost=114.47..115.09 rows=250 width=17)
   Sort Key: e.name
   ->  Hash Join  (cost=1.26..104.51 rows=250 width=17)
         Hash Cond: (e.dept_id = d.id)
         ->  Seq Scan on employee e  (cost=0.00..82.00 rows=5000 width=21)
         ->  Hash  (cost=1.25..1.25 rows=1 width=4)
               ->  Seq Scan on dept d  (cost=0.00..1.25 rows=1 width=4)
                     Filter: ((name)::text = 'Engineering'::text)
(8 rows)


EXPLAIN ANALYZE SELECT e.id, e.name FROM employee e JOIN dept d ON e.dept_id = d.id WHERE d.name='Engineering' ORDER BY e.name;
                                                       QUERY PLAN
------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=114.47..115.09 rows=250 width=17) (actual time=1.389..1.411 rows=276 loops=1)
   Sort Key: e.name
   Sort Method: quicksort  Memory: 37kB
   ->  Hash Join  (cost=1.26..104.51 rows=250 width=17) (actual time=0.085..1.244 rows=276 loops=1)
         Hash Cond: (e.dept_id = d.id)
         ->  Seq Scan on employee e  (cost=0.00..82.00 rows=5000 width=21) (actual time=0.028..0.512 rows=5000 loops=1)
         ->  Hash  (cost=1.25..1.25 rows=1 width=4) (actual time=0.036..0.037 rows=1 loops=1)
               Buckets: 1024  Batches: 1  Memory Usage: 9kB
               ->  Seq Scan on dept d  (cost=0.00..1.25 rows=1 width=4) (actual time=0.021..0.023 rows=1 loops=1)
                     Filter: ((name)::text = 'Engineering'::text)
                     Rows Removed by Filter: 19
 Planning Time: 0.315 ms
 Execution Time: 1.486 ms
(13 rows)
```

Amr Elhelw's
TECH
VAULT



```
EXPLAIN SELECT e.id, e.name FROM employee e JOIN dept d ON e.dept_id = d.id JOIN location l ON d.loc_id = l.id WHERE l.city='London'
ORDER BY e.name;
                                       QUERY PLAN
----------------------------------------------------------------------------------------
 Sort  (cost=232.95..237.95 rows=2000 width=17)
   Sort Key: e.name
   ->  Hash Join  (cost=2.54..123.29 rows=2000 width=17)
         Hash Cond: (e.dept_id = d.id)
         ->  Seq Scan on employee e  (cost=0.00..82.00 rows=5000 width=21)
         ->  Hash  (cost=2.44..2.44 rows=8 width=4)
               ->  Hash Join  (cost=1.09..2.44 rows=8 width=4)
                     Hash Cond: (d.loc_id = l.id)
                     ->  Seq Scan on dept d  (cost=0.00..1.20 rows=20 width=8)
                     ->  Hash  (cost=1.06..1.06 rows=2 width=4)
                           ->  Seq Scan on location l  (cost=0.00..1.06 rows=2 width=4)
                                 Filter: ((city)::text = 'London'::text)
(12 rows)
```

https://www.pgexplain.dev/



Plan

```
                    QUERY PLAN
--------------------------------------------------------------------------------
 Sort  (cost=232.95..237.95 rows=2000 width=17)
   Sort Key: e.name
   ->  Hash Join  (cost=2.5
         Hash Cond: (e.dept_
         ->  Seq Scan on emp
         ->  Hash  (cost=2.44
               ->  Hash Join  (c
                     Hash Cond: (
                     ->  Seq Scan
                     ->  Hash  (co
                           ->  Seq S
                                 Filter:
```

Execution time: N/A Planning time: N/A Triggers: N/A

```
#1 Sort
#2 └ Hash Join
#3  ├ Seq Scan
#4  └ Hash
#5   └ Hash Join
#6    ├ Seq Scan
#7    └ Hash
#8     └ Seq Scan
```

#1 **Sort** $
by e.name

#2 **Hash Join** $
on e.dept_id = d.id

#3 **Seq Scan** $
on employee as e

#4 **Hash**

#5 **Hash Join**
on d.loc_id = l.id

#6 **Seq Scan**
on dept as d

#7 **Hash**

#8 **Seq Scan**
on location as l