aiven

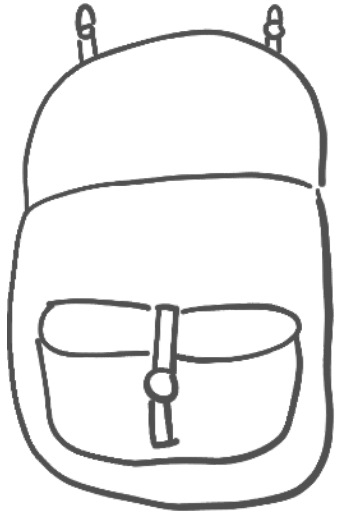
Solving the Knapsack Problem with Recursive Queries and PostgreSQL

Francesco Tisiot - @ftisiot - Developer Advocate

Knapsack Problem

20⚖️

10❤️- 3⚖️



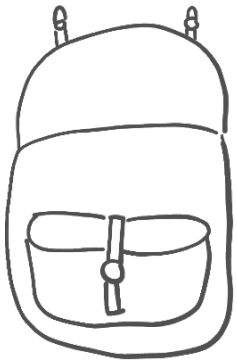
15❤️- 5⚖️

5❤️- 15⚖️

8❤️- 10⚖️

7❤️- 10⚖️

20⚖️

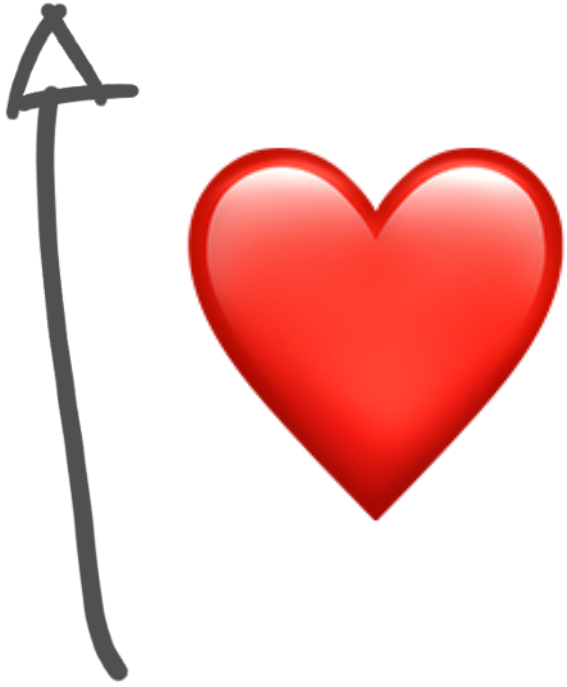


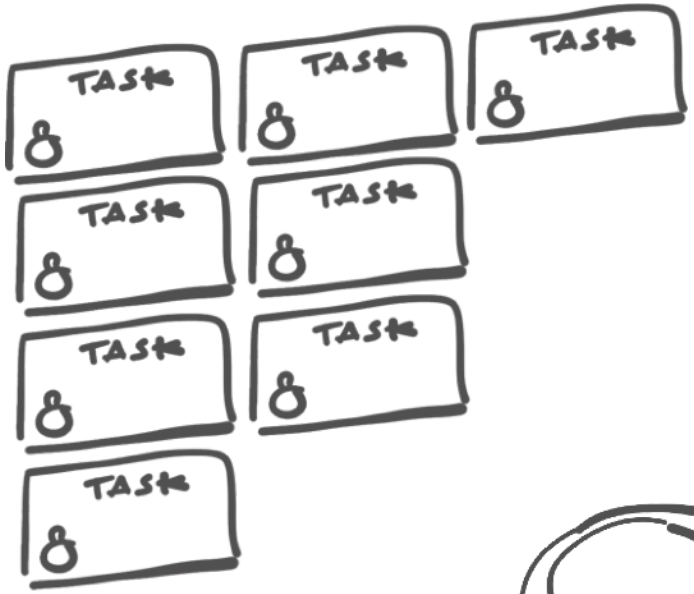
20❤️ - 20⚖️

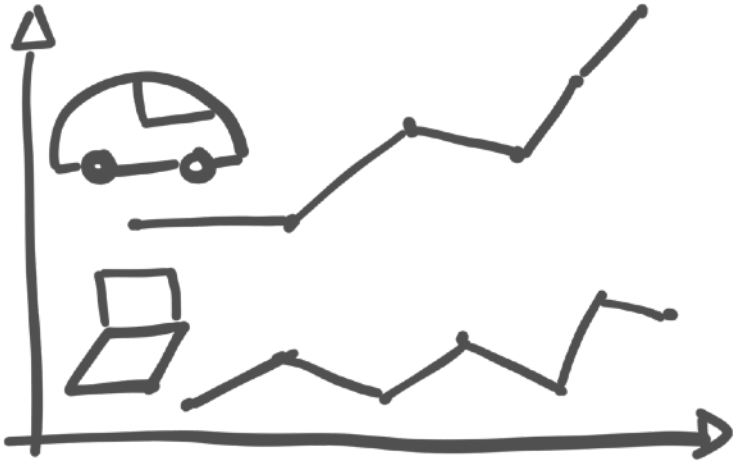
5❤️- 15⚖️ 15❤️- 5⚖️

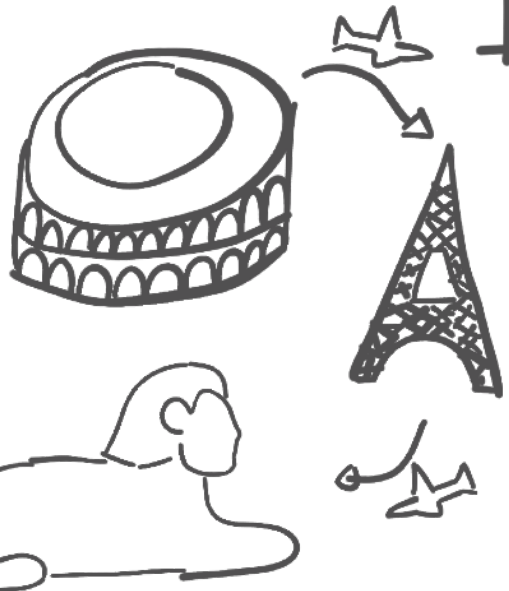
32❤️ - 18⚖️

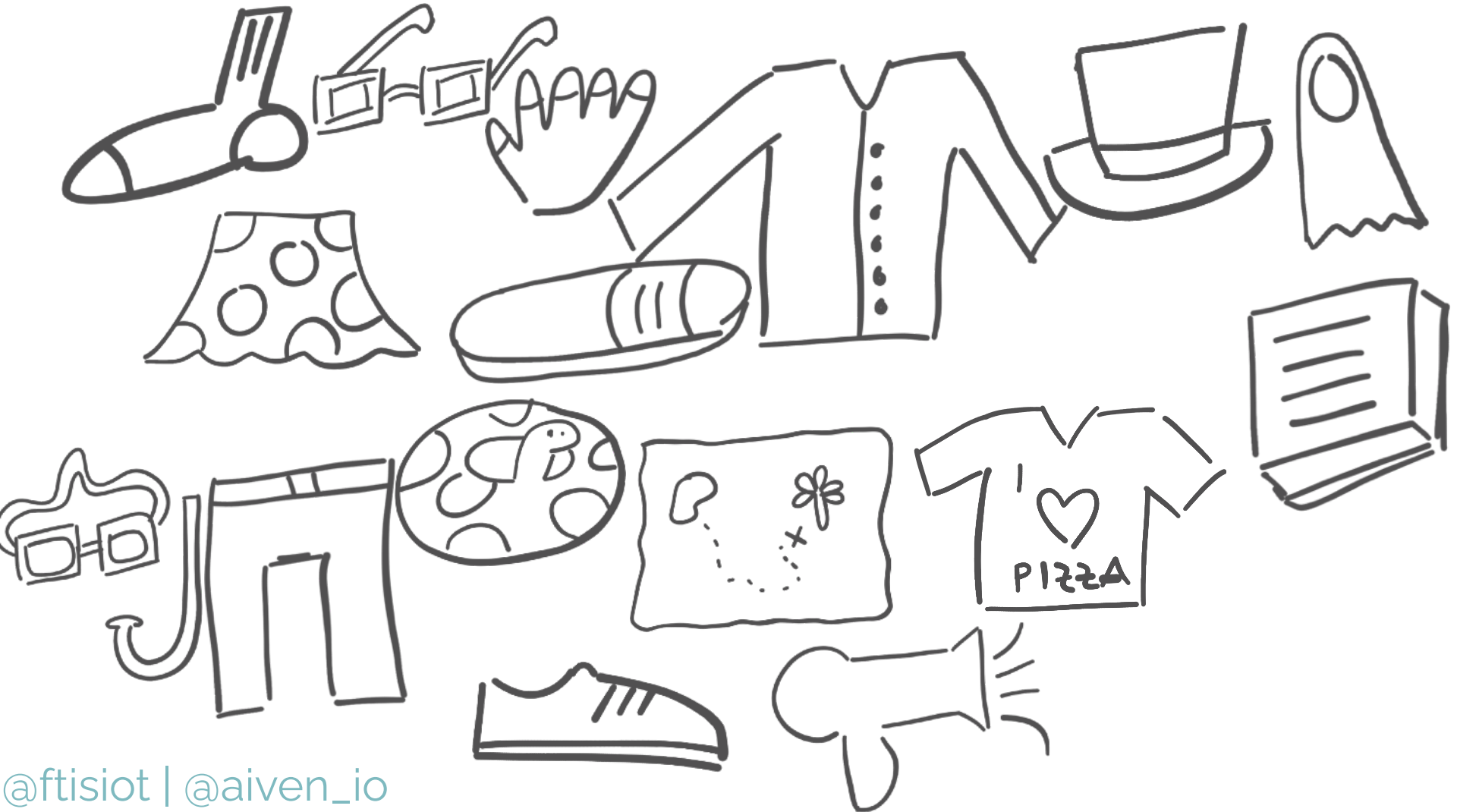
10❤️- 3⚖️ 7❤️- 10⚖️ 15❤️- 5⚖️




TASK
TASK
TASK
TASK
TASK
TASK
TASK
TASK



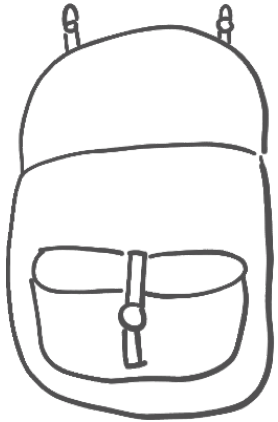



I ♡
PIZZA

10❤️ - 5 💚

3⚖️ - 1🧊

20⚖️ - 30🧊



8❤️ - 3 💚

10⚖️ - 9🧊

15❤️ - 5 💚

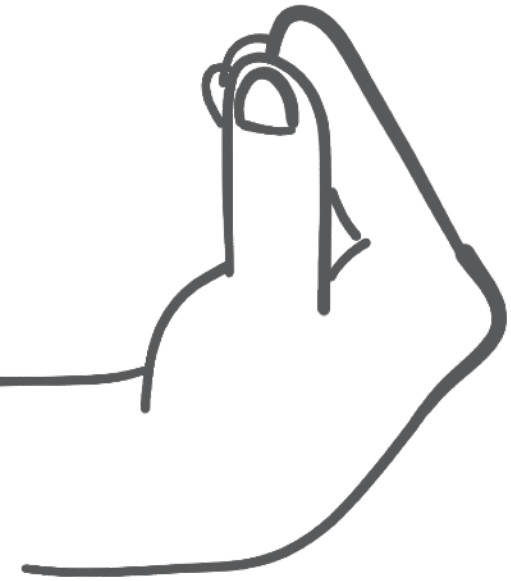
5⚖️ - 10🧊

7❤️ - 15 💚

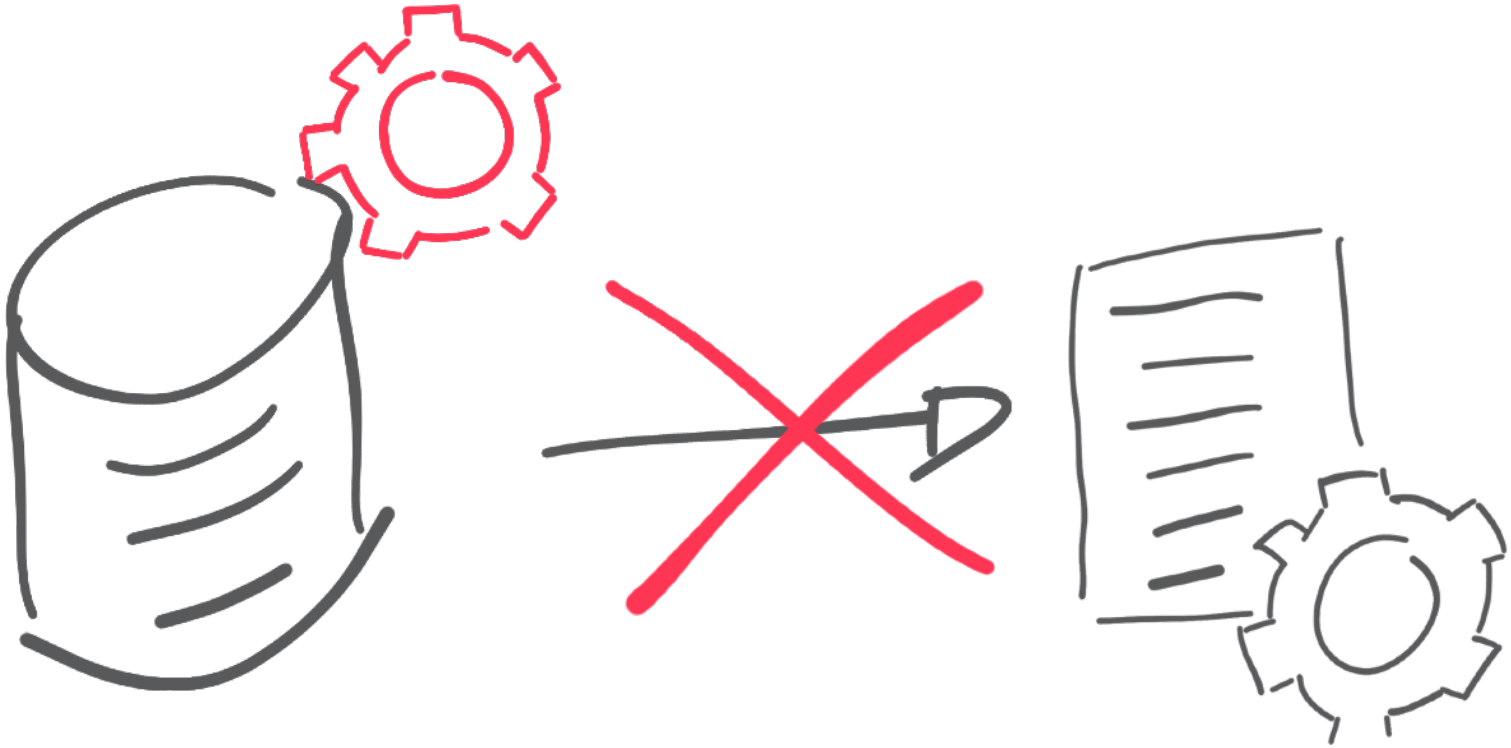
10⚖️ - 5🧊

5❤️ - 5 💚

15⚖️ - 10🧊

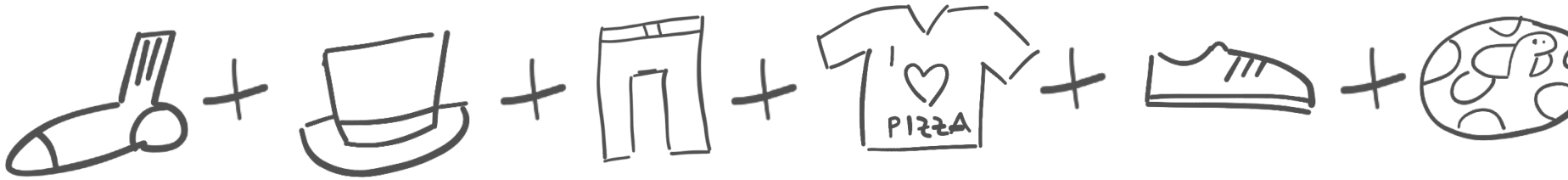


Why in PostgreSQL?



Recursive Queries

Variable number of iterations



```
create table inventory (
    item_id integer,
    item_name varchar,
    value int,
    weight int);

insert into inventory values
    (1,'Socks'   , 10,  3),
    (2,'Hat'     , 15,  5),
    (3,'Trousers',  5, 15),
    (4,'Shoes'   ,  8, 10),
    (5,'T-Shirt' ,  7, 10);
```

Item	Value	Weight
Socks	10	3
Hat	15	5
Trousers	5	15
Shoes	8	10
T-Shirt	7	10

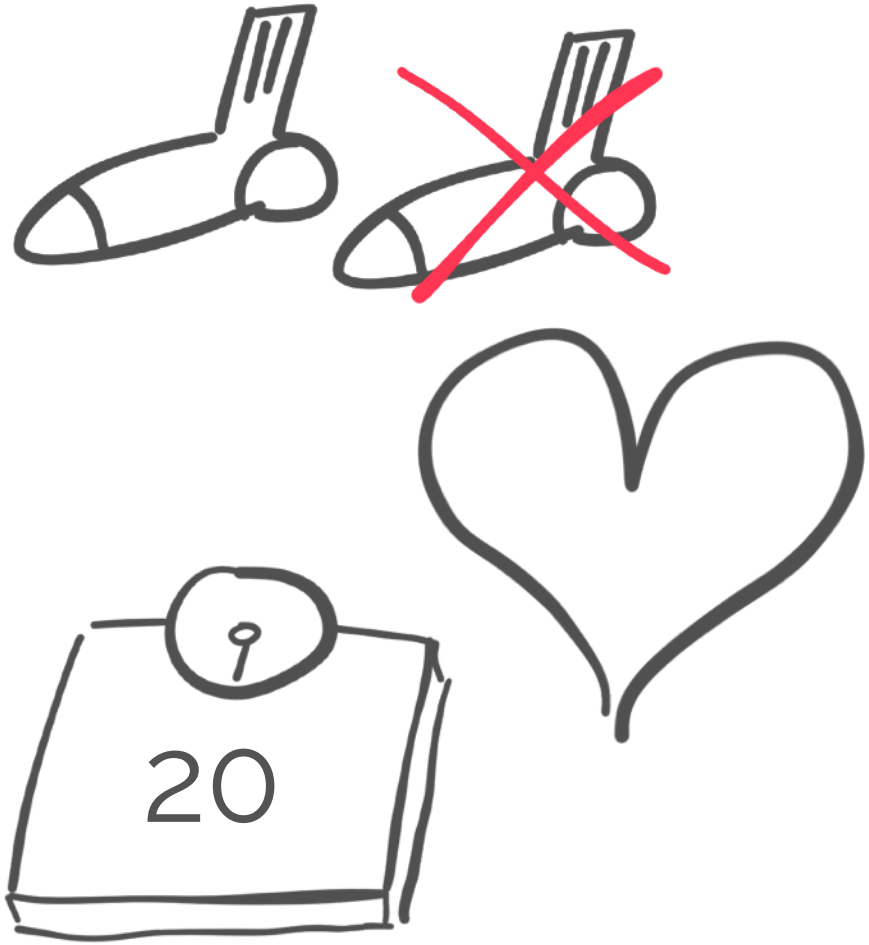
Recursive Queries - How To

@ftisiot | @aiven_io

WITH RECURSIVE items(picked_items, nr_items) as (
SELECT ARRAY[item_name] as picked_items,
1 nr_items
from inventory where item_name = 'Socks'
UNION ALL
select picked_items || item_name,
nr_items + 1
from inventory
cross join items
where nr_items+1 <= 3
)
select * from items where nr_items=3;

```
                name                 | nr_items
-------------------------------------+----------
 {Socks,Socks,Socks}                 |        3
 {Socks,Socks,Hat}                   |        3
 {Socks,Socks,Trousers}              |        3
 {Socks,Socks,Shoes}                 |        3
 {Socks,Socks,T-Shirt}               |        3
 ...
 {Socks,T-Shirt,Hat}                 |        3
 {Socks,T-Shirt,Trousers}            |        3
 {Socks,T-Shirt,Shoes}               |        3
 {Socks,T-Shirt,T-Shirt}             |        3
(25 rows)
```

Knapsack Constraints



```
WITH RECURSIVE items(item_id, picked_items, nr_items, total_weight, total_value) as (
    SELECT
        item_id,
        ARRAY[item_name] as picked_items,
        1 nr_items,
        weight total_weight,
        value total_value
    from inventory
    UNION ALL
    select
        inventory.item_id,
        picked_items || item_name,
        nr_items + 1,
        weight + total_weight,
        value + total_value
    from inventory cross join items
    where
        picked_items::varchar[] @> ARRAY[item_name] = false
        and weight + total_weight <= 20

)
select * from items order by total_value;
```

item_id	picked_items	nr_items	tot_weight	tot_value
3	{Trousers}	1	15	5
5	{T-Shirt}	1	10	7
4	{Shoes}	1	10	8
1	{Socks}	1	3	10
5	{Shoes,T-Shirt}	2	20	15
4	{T-Shirt,Shoes}	2	20	15
. . .				
2	{Socks,T-Shirt,Hat}	3	18	32
5	{Hat,Socks,T-Shirt}	3	18	32
1	{Hat,T-Shirt,Socks}	3	18	32
2	{T-Shirt,Socks,Hat}	3	18	32
1	{Hat,Shoes,Socks}	3	18	33
4	{Socks,Hat,Shoes}	3	18	33
2	{Socks,Shoes,Hat}	3	18	33
2	{Shoes,Socks,Hat}	3	18	33
4	{Hat,Socks,Shoes}	3	18	33
1	{Shoes,Hat,Socks}	3	18	33

(33 rows)

~~{Socks,Shoes,Hat}~~

~~{Socks,Hat,Shoes}~~

~~{Shoes,Socks,Hat}~~

~~{Shoes,Hat,Socks}~~

~~{Hat,Shoes,Socks}~~

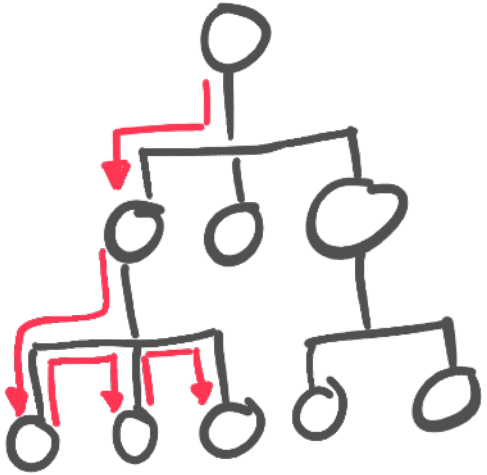
{Hat,Socks,Shoes}

```
UNION ALL
select
    inventory.item_id,
    picked_items || item_name,
    nr_items + 1,
    weight + total_weight,
    value + total_value
from inventory cross join items
where
    picked_items::varchar[] @> ARRAY[item_name] = false
    and weight + total_weight <= 20
    and inventory.item_id > items.item_id
```

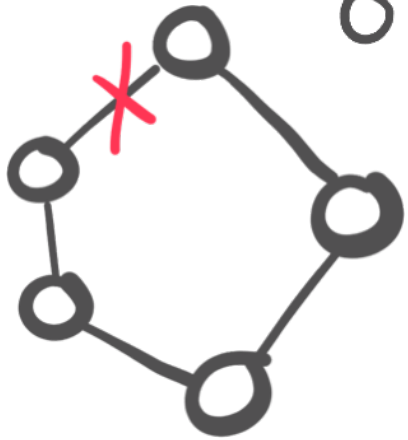
item_id	picked_items	nr_items	tot_weight	tot_value
3	{Trousers}	1	15	5
5	{T-Shirt}	1	10	7
4	{Shoes}	1	10	8
1	{Socks}	1	3	10
2	{Hat}	1	5	15
5	{Shoes,T-Shirt}	2	20	15
3	{Socks,Trousers}	2	18	15
5	{Socks,T-Shirt}	2	13	17
4	{Socks,Shoes}	2	13	18
3	{Hat,Trousers}	2	20	20
5	{Hat,T-Shirt}	2	15	22
4	{Hat,Shoes}	2	15	23
2	{Socks,Hat}	2	8	25
5	{Socks,Hat,T-Shirt}	3	18	32
4	{Socks,Hat,Shoes}	3	18	33

(15 rows)

New in Pg 14



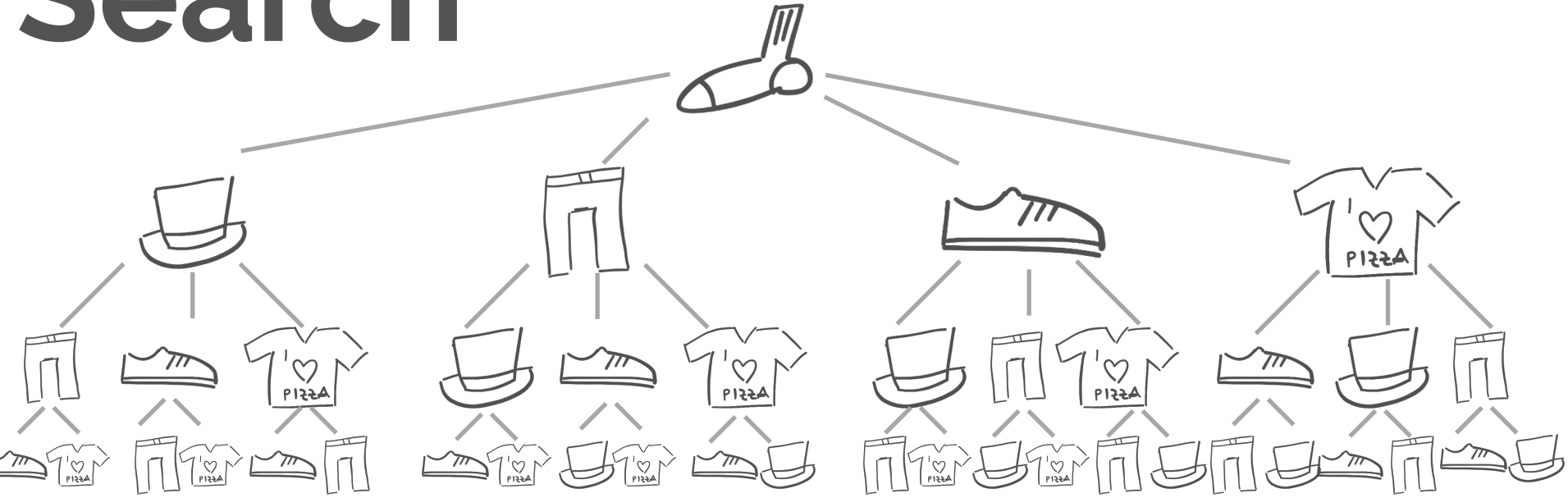
SEARCH



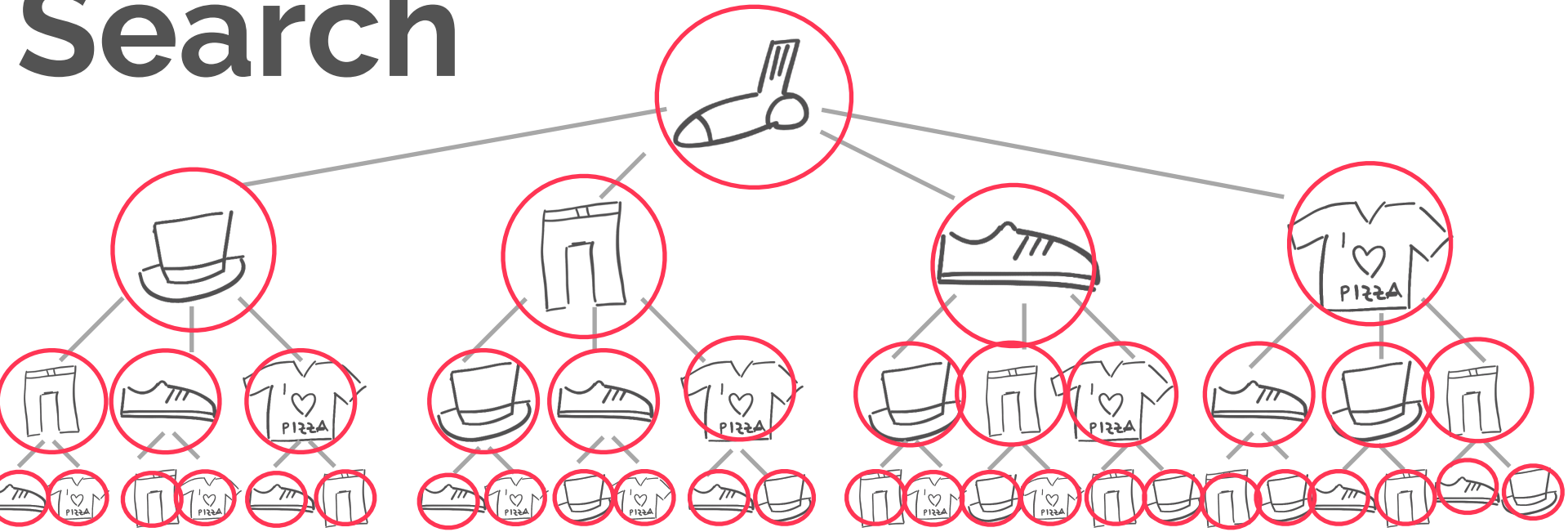
CYCLE

@ftisiot | @aiven_io

Search



Search



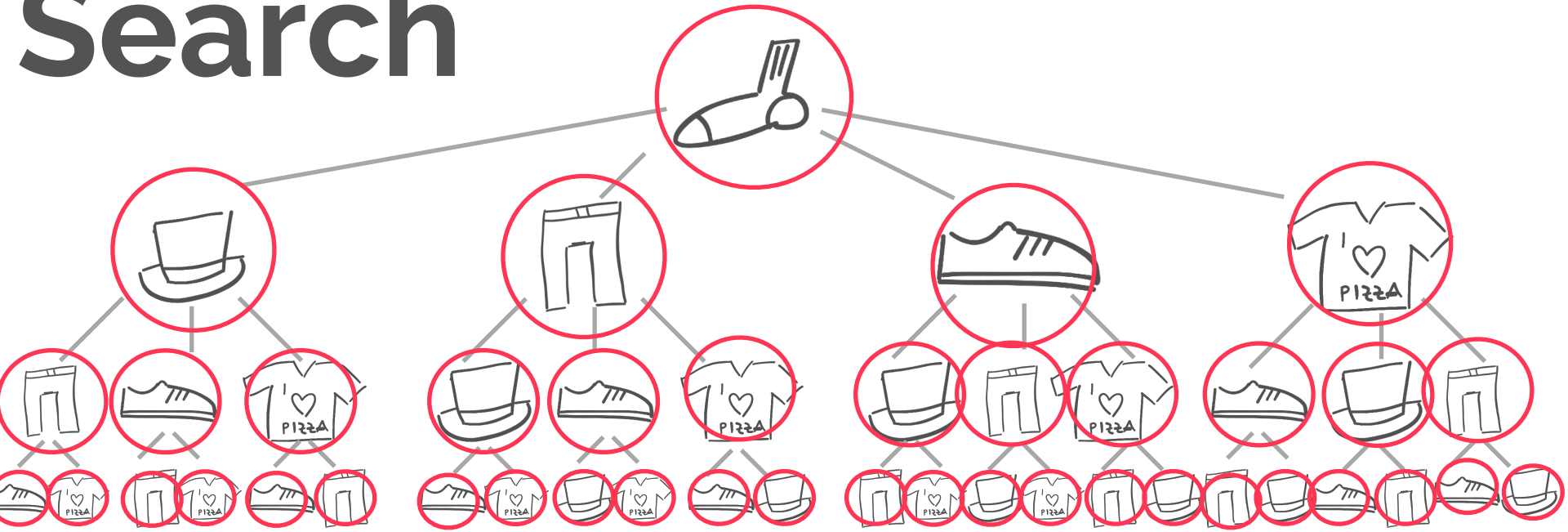
BREADTH

```
WITH RECURSIVE items(item_id, picked_items, nr_items, total_weight, total_value) as (
    SELECT
        item_id,
        ARRAY[item_name] as picked_items,
        1 nr_items,
        weight total_weight,
        value total_value
    from inventory
    UNION ALL
    select
        inventory.item_id,
        picked_items || item_name,
        nr_items + 1,
        weight + total_weight,
        value + total_value
    from inventory cross join items
    where
        picked_items::varchar[] @> ARRAY[item_name] = false
        and weight + total_weight <= 20

) SEARCH BREADTH FIRST BY item_id SET ordercol
select * from items order by ordercol;
```

item_id	picked_items	nr_items	tot_weight	tot_value	ordercol
1	{Socks}	1	3	10	(0,1)
2	{Socks,Hat}	2	8	25	(1,2)
3	{Socks,Trousers}	2	18	15	(1,3)
4	{Socks,Shoes}	2	13	18	(1,4)
5	{Socks,T-Shirt}	2	13	17	(1,5)
2	{Socks,Trousers,Hat}	3	23	30	(2,2)
2	{Socks,Shoes,Hat}	3	18	33	(2,2)
2	{Socks,T-Shirt,Hat}	3	18	32	(2,2)
3	{Socks,Hat,Trousers}	3	23	30	(2,3)
3	{Socks,T-Shirt,Trousers}	3	28	22	(2,3)
3	{Socks,Shoes,Trousers}	3	28	23	(2,3)
4	{Socks,Hat,Shoes}	3	18	33	(2,4)
4	{Socks,T-Shirt,Shoes}	3	23	25	(2,4)
4	{Socks,Trousers,Shoes}	3	28	23	(2,4)
5	{Socks,Hat,T-Shirt}	3	18	32	(2,5)
5	{Socks,Shoes,T-Shirt}	3	23	25	(2,5)
5	{Socks,Trousers,T-Shirt}	3	28	22	(2,5)

(17 rows)

Search



DEPTH

item_id	picked_items	nr_items	tot_weight	tot_value	ordercol
1	{Socks}	1	3	10	{(1)}
2	{Socks,Hat}	2	8	25	{(1),(2)}
3	{Socks,Hat,Trousers}	3	23	30	{(1),(2),(3)}
4	{Socks,Hat,Shoes}	3	18	33	{(1),(2),(4)}
5	{Socks,Hat,T-Shirt}	3	18	32	{(1),(2),(5)}
3	{Socks,Trousers}	2	18	15	{(1),(3)}
2	{Socks,Trousers,Hat}	3	23	30	{(1),(3),(2)}
4	{Socks,Trousers,Shoes}	3	28	23	{(1),(3),(4)}
5	{Socks,Trousers,T-Shirt}	3	28	22	{(1),(3),(5)}
4	{Socks,Shoes}	2	13	18	{(1),(4)}
2	{Socks,Shoes,Hat}	3	18	33	{(1),(4),(2)}
3	{Socks,Shoes,Trousers}	3	28	23	{(1),(4),(3)}
5	{Socks,Shoes,T-Shirt}	3	23	25	{(1),(4),(5)}
5	{Socks,T-Shirt}	2	13	17	{(1),(5)}
2	{Socks,T-Shirt,Hat}	3	18	32	{(1),(5),(2)}
3	{Socks,T-Shirt,Trousers}	3	28	22	{(1),(5),(3)}
4	{Socks,T-Shirt,Shoes}	3	23	25	{(1),(5),(4)}

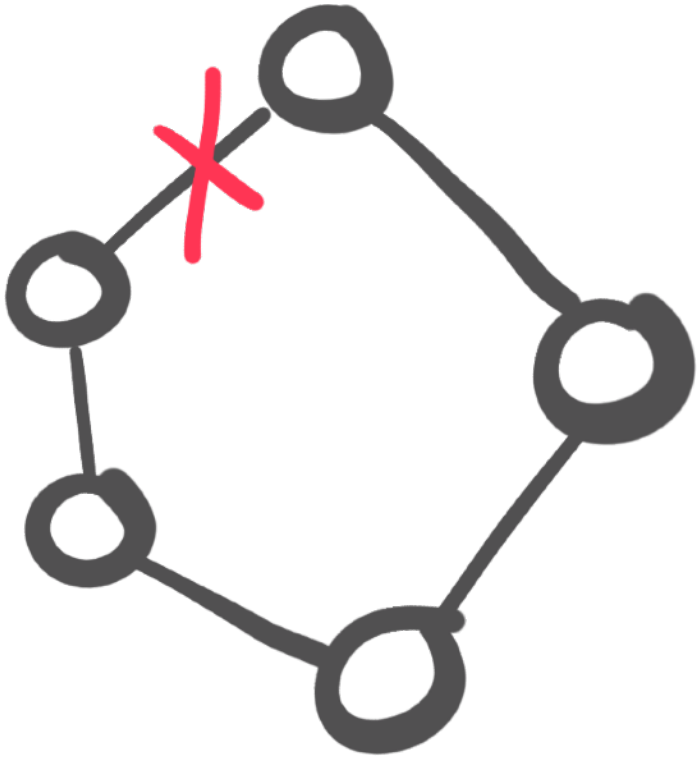
(17 rows)

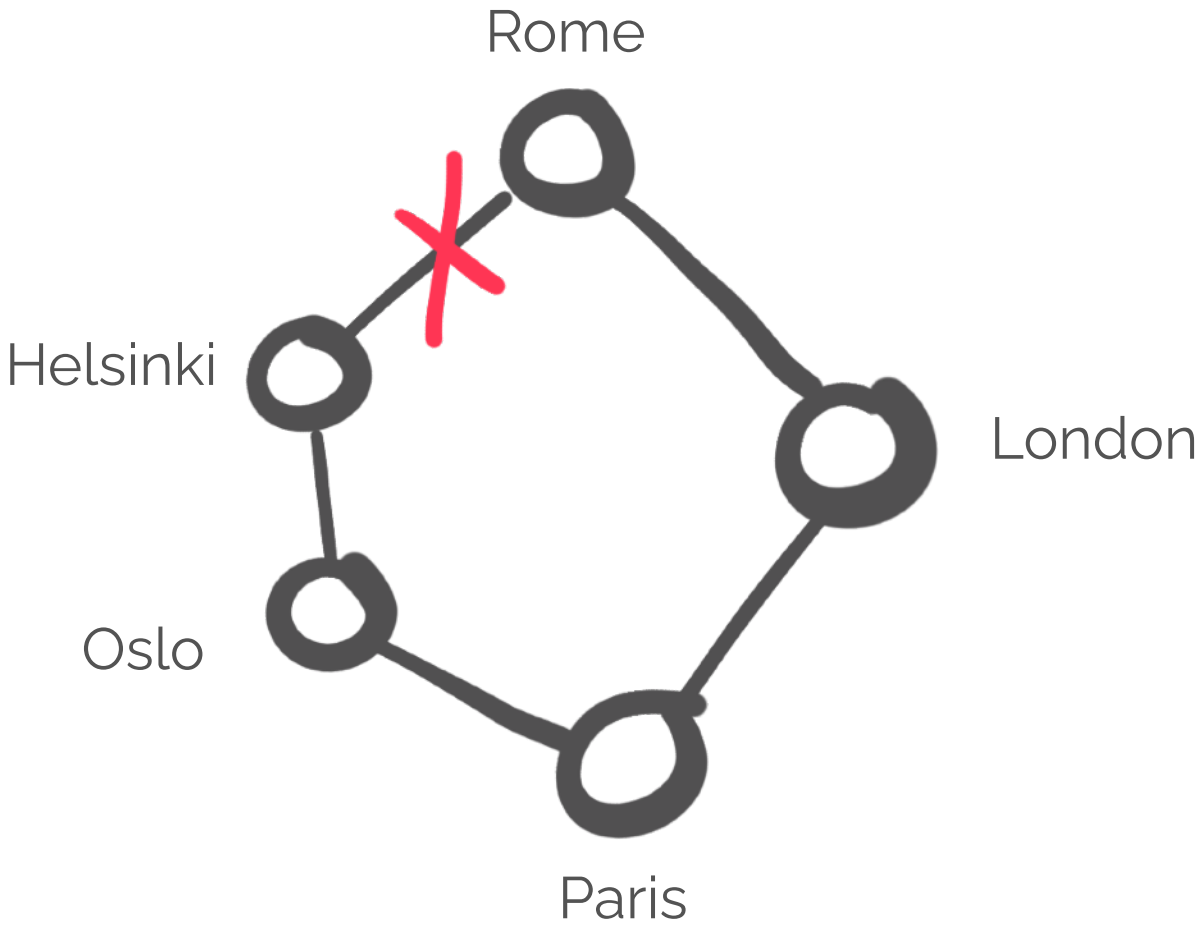
```
WITH RECURSIVE items(item_id, picked_items, nr_items, total_weight, total_value) as (
    SELECT
        item_id,
        ARRAY[item_name] as picked_items,
        1 nr_items,
        weight total_weight,
        value total_value
    from inventory
    UNION ALL
    select
        inventory.item_id,
        picked_items || item_name,
        nr_items + 1,
        weight + total_weight,
        value + total_value
    from inventory cross join items
    where
        picked_items::varchar[] @> ARRAY[item_name] = false
        and weight + total_weight <= 20

) SEARCH DEPTH FIRST BY item_id SET ordercol
select * from items order by ordercol;
```

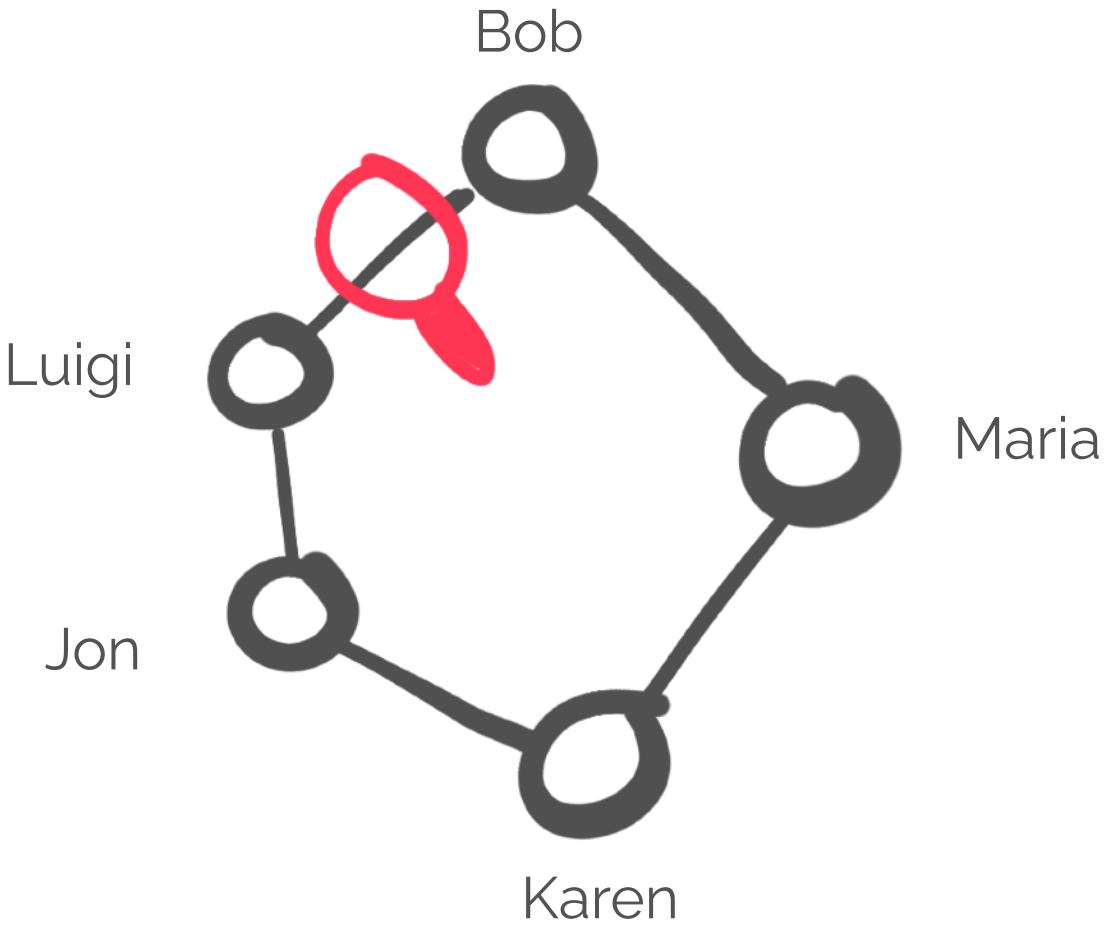
item_id	picked_items	nr_items	tot_weight	tot_value	ordercol
1	{Socks}	1	3	10	(0,1)
2	{Socks,Hat}	2	8	25	(1,2)
3	{Socks,Trousers}	2	18	15	(1,3)
4	{Socks,Shoes}	2	13	18	(1,4)
5	{Socks,T-Shirt}	2	13	17	(1,5)
2	{Socks,Trousers,Hat}	3	23	30	(2,2)
2	{Socks,Shoes,Hat}	3	18	33	(2,2)
2	{Socks,T-Shirt,Hat}	3	18	32	(2,2)
3	{Socks,Hat,Trousers}	3	23	30	(2,3)
3	{Socks,T-Shirt,Trousers}	3	28	22	(2,3)

item_id	picked_items	nr_items	tot_weight	tot_value	ordercol
1	{Socks}	1	3	10	{(1)}
2	{Socks,Hat}	2	8	25	{(1),(2)}
3	{Socks,Hat,Trousers}	3	23	30	{(1),(2),(3)}
4	{Socks,Hat,Shoes}	3	18	33	{(1),(2),(4)}
5	{Socks,Hat,T-Shirt}	3	18	32	{(1),(2),(5)}
3	{Socks,Trousers}	2	18	15	{(1),(3)}
2	{Socks,Trousers,Hat}	3	23	30	{(1),(3),(2)}
4	{Socks,Trousers,Shoes}	3	28	23	{(1),(3),(4)}
5	{Socks,Trousers,T-Shirt}	3	28	22	{(1),(3),(5)}
4	{Socks,Shoes}	2	13	18	{(1),(4)}
2	{Socks,Shoes,Hat}	3	18	33	{(1),(4),(2)}

Cycle




Rome
Helsinki
London
Oslo
Paris


Bob
Luigi
Maria
Jon
Karen

```
picked_items::varchar[] @> ARRAY[item_name] = false
```

```
inventory.item_id > items.item_id
```

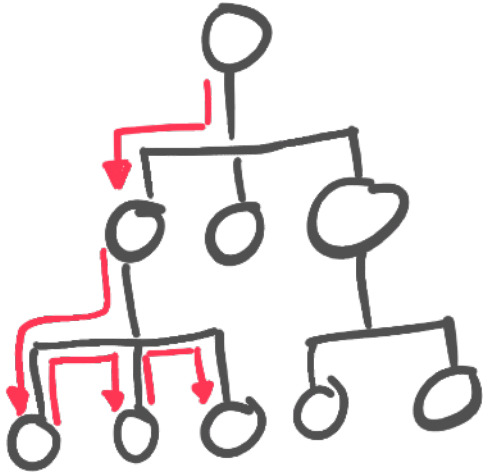
```
CYCLE item_id SET is_cycle USING items_ids
```

item_id	picked_items	is_cycle	items_ids
1	{Socks}	f	{(1)}
1	{Socks,Socks}	t	{(1),(1)}
2	{Socks,Hat}	f	{(1),(2)}
3	{Socks,Trousers}	f	{(1),(3)}
4	{Socks,Shoes}	f	{(1),(4)}
5	{Socks,T-Shirt}	f	{(1),(5)}
1	{Socks,Hat,Socks}	t	{(1),(2),(1)}
2	{Socks,Hat,Hat}	t	{(1),(2),(2)}
3	{Socks,Hat,Trousers}	f	{(1),(2),(3)}
4	{Socks,Hat,Shoes}	f	{(1),(2),(4)}
5	{Socks,Hat,T-Shirt}	f	{(1),(2),(5)}
1	{Socks,Trousers,Socks}	t	{(1),(3),(1)}

. . .

Knapsack Problem

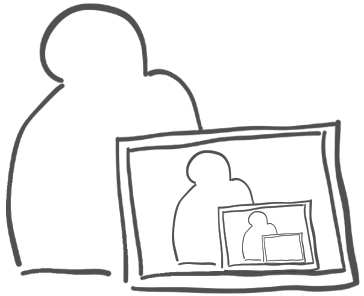
Base + Recursion

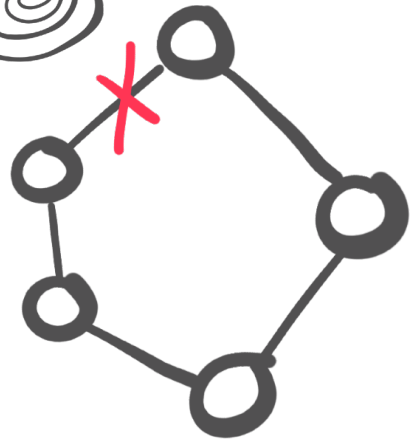


SEARCH

PostgreSQL

Recursive Queries





CYCLE

References

Knapsack Problem

https://en.wikipedia.org/wiki/Knapsack_problem

Knapsack in PostgreSQL

https://aiven.io/blog/solving-the-knapsack-problem-in-postgresql

PostgreSQL 14 Search and Cycle Features

https://aiven.io/blog/explore-the-new-search-and-cycle-features-in-postgresql-14

Aiven 

https://aiven.io/