

The image shows a low-angle shot of a modern glass skyscraper. The building's facade is composed of large glass panels held together by a dark metal grid. The glass reflects a vibrant sky with blue, purple, and green hues, suggesting a sunset or sunrise. Centered on the facade is the J.P. Morgan logo, which consists of the letters 'J.P. Morgan' in a white, three-dimensional, serif font. The letters are slightly shadowed, giving them a sense of depth as they appear to be attached to the building.

J.P. Morgan

## SENIOR DATA ENGINEER - SQL QUESTION

Could you provide a SQL query that pairs the eldest adults from the 'family' table with the youngest children? Additionally, could you explain the logic behind this query and how it achieves this pairing?

```
-- Create the 'family' table
CREATE TABLE family (
  person VARCHAR(5),
  type VARCHAR(10),
  age INT
);

-- Insert data into the 'family' table
INSERT INTO family VALUES
  ('A1', 'Adult', 54),
  ('A2', 'Adult', 53),
  ('A3', 'Adult', 52),
  ('A4', 'Adult', 58),
  ('A5', 'Adult', 54),
  ('C1', 'Child', 20),
  ('C2', 'Child', 19),
  ('C3', 'Child', 22),
  ('C4', 'Child', 15);
```



```
WITH cte_adult AS (  
    SELECT  
        person, age, type,  
        ROW_NUMBER() OVER (ORDER BY age DESC) AS rnk  
    FROM  
        family  
    WHERE  
        type = 'Adult'  
)  
  
cte_child AS (  
    SELECT  
        person, age, type,  
        ROW_NUMBER() OVER (ORDER BY age) AS rnk  
    FROM  
        family  
    WHERE  
        type = 'Child'  
)  
  
SELECT  
    a.person AS adult_person,  
    c.person AS child_person  
FROM  
    cte_adult a  
LEFT JOIN  
    cte_child c  
USING (rnk);
```

## SENIOR DATA ENGINEER - EXPLANATION

This SQL code uses common table expressions (CTEs) to create two separate lists of individuals, one for adults and one for children, ranking them by age in descending order for adults and ascending order for children.

It then performs a left join between these CTEs based on their age rankings, effectively pairing the eldest adults with the youngest children.

The result is a set of pairs, where each pair consists of one adult and one child with a similar age ranking, showcasing a creative approach to organizing and analyzing family data in order to emphasize age-based pairings.



## Schema SQL ●

```
1 create table family
2 (
3   person varchar(5),
4   type varchar(10),
5   age int
6 );
7 insert into family values
8 ('A1','Adult',54)
9 ,('A2','Adult',53)
10 ,('A3','Adult',52)
11 ,('A4','Adult',58)
12 ,('A5','Adult',54)
13 ,('C1','child',20)
14 ,('C2','child',19)
15 ,('C3','child',22)
16 ,('C4','child',15);
17
```

## Query SQL ●

```
1 #select * from family;
2 WITH cte_adult AS (
3   SELECT
4     person,
5     age,
6     type,
7     ROW_NUMBER() OVER (ORDER BY age DESC) AS rnk
8   FROM
9     family
10  WHERE
11    type = 'Adult'
12 ),
13 cte_child AS (
14   SELECT
15     person,
16     age,
17     type,
18     ROW_NUMBER() OVER (ORDER BY age) AS rnk
19   FROM
20     family
21  WHERE
22    type = 'child'
23 )
24
25 SELECT
26   a.person AS adult_person,
27   c.person AS child_person
28 FROM
29   cte_adult a
30 LEFT JOIN
31   cte_child c
32 USING (rnk);
33
```

Running query...



Results

Copy as Markdown ↗

Query #1 Execution time: 10ms

adult_person	child_person
A4	C4
A1	C2
A5	C1
A2	C3
A3	null