Interactive - 22 - recursive select - populating existing tables

It is also possible to store and query recursive data. You need a table with and "id" and a "parent_id" that referees back to the same table.

```
DROP TABLE if exists tree_example ;

CREATE TABLE tree_example (
    id              int not null primary key,
    manager_id      int,
    name            text
);

INSERT INTO tree_example ( id, name, manager_id ) values
    (1, 'Michael North', NULL),
    (2, 'Megan Berry', 1),
    (3, 'Sarah Berry', 1),
    (4, 'Zoe Black', 1),
    (5, 'Tim James', 1),
    (6, 'Bella Tucker', 2),
    (7, 'Ryan Metcalfe', 2),
    (8, 'Max Mills', 2),
    (9, 'Benjamin Glover', 2),
    (10, 'Carolyn Henderson', 3),
    (11, 'Nicola Kelly', 3),
    (12, 'Alexandra Climo', 3),
    (13, 'Dominic King', 3),
    (14, 'Leonard Gray', 4),
    (15, 'Eric Rampling', 4),
    (16, 'Piers Paige', 7),
    (17, 'Ryan Henderson', 7),
    (18, 'Frank Tucker', 8),
    (19, 'Nathan Ferguson', 8),
    (20, 'Kevin Rampling', 8)
;
```

Now we can to a recursive query. In PostgreSQL they are composed of a base query unironed with all the children.

```
WITH RECURSIVE subordinates AS (
        SELECT
            id,
            manager_id,
            name
        FROM tree_example
```

```
        WHERE id = 2
    UNION
        SELECT
            e.id,
            e.manager_id,
            e.name
        FROM tree_example e
        INNER JOIN subordinates s ON s.id = e.manager_id
) SELECT * FROM subordinates
;
```