Using self joins to pull hierarchical relationships

Business problem: Pulling employee/manager data with a self join

The VP of sales is currently contacting all of the managers who have direct reports in the Sales department to notify them of the new commission structure. Using the `employees` table, which contains all employees and their associated manager, you can pull a report of all sales employees and their manager's email address using a self join. However, with your sharp thinking, you notice that several employees have `null` values for their manager's email address due to the fact that some employees don't have a manager logged in the database.
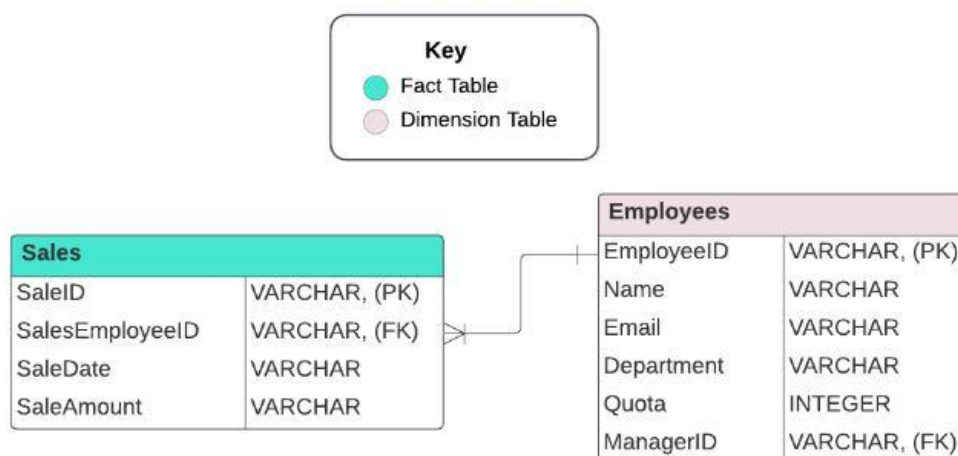
Considering that the goal of this business problem is to spread the word about an important business change that will affect the sales team, you know that `null` email addresses won't be useful in the output. I mean...how would *you* feel if you weren't notified of this important change because you didn't have a manager logged for you in the database? We can't always accept `null` values, and sometimes we have to recode them or *impute* them with another value that is more meaningful for the business. Without bothering the super busy VP with this, you take the initiative and modify your query so that it replaces all `null` manager emails with the employees email. This way, all sales employees without a manager email will receive the information directly.

Task:

Create an email list from the `employees` table that includes the following columns for all employees under `department = 'Sales'`.

- `employeeid`
- `employee_name`
- `manager_name`
- `contact_email` (use the manager email if available and the employee email if not)

Sales Data Model

Key
- Fact Table
- Dimension Table

Sales

SaleID	VARCHAR, (PK)
SalesEmployeeID	VARCHAR, (FK)
SaleDate	VARCHAR
SaleAmount	VARCHAR

Employees

EmployeeID	VARCHAR, (PK)
Name	VARCHAR
Email	VARCHAR
Department	VARCHAR
Quota	INTEGER
ManagerID	VARCHAR, (FK)

```sql
1    -- SQL code by John Uzoma
2    SELECT
3        E.employeeid,
4        E.name as employee_name,
5        M.name as manager_name,
6        COALESCE(M.email, E.email) as contact_email
7    FROM EMPLOYEES E
8    LEFT JOIN EMPLOYEES M
9    ON E.managerid = M.employeeid
10   WHERE E.department = 'Sales';
11
```

Result

```
----------------------------------------------------------------------
| EMPLOYEEID | EMPLOYEE_NAME | MANAGER_NAME | CONTACT_EMAIL           |
----------------------------------------------------------------------
| E738       | Joe Burns     | Bonnie Clark | bclark6@bing.com        |
| E429       | Bonnie Clark  | null         | bclark6@bing.com        |
| E192       | Lori Fields   | Bonnie Clark | bclark6@bing.com        |
| E172       | Roy Martin    | null         | rmarting@delicious.com  |
----------------------------------------------------------------------
```