Table: Products

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| product_id  | int     |
| store1      | int     |
| store2      | int     |
| store3      | int     |
+-------------+---------+
product_id is the primary key for this table.
Each row in this table indicates the product&#39;s price in 3 different stores: store1, sto
If the product is not available in a store, the price will be null in that store&#39;s colu
```

Write an SQL query to rearrange the Products table so that each row has (product_id, store, price). If a product is not available in a store, do **not** include a row with that product_id and store combination in the result table.

Return the result table in **any order**.

The query result format is in the following example.

Example 1:**

```
Input:
Products table:
+------------+--------+--------+--------+
| product_id | store1 | store2 | store3 |
+------------+--------+--------+--------+
| 0          | 95     | 100    | 105    |
| 1          | 70     | null   | 80     |
+------------+--------+--------+--------+
Output:
+------------+--------+-------+
| product_id | store  | price |
+------------+--------+-------+
| 0          | store1 | 95    |
| 0          | store2 | 100   |
| 0          | store3 | 105   |
| 1          | store1 | 70    |
| 1          | store3 | 80    |
+------------+--------+-------+
Explanation:
Product 0 is available in all three stores with prices 95, 100, and 105 respectively.
Product 1 is available in store1 with price 70 and store3 with price 80. The product is not
```