Table: `Genders`

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| user_id     | int     |
| gender      | varchar |
+-------------+---------+
user_id is the primary key for this table.
gender is ENUM of type 'female', 'male', or 'other'.
Each row in this table contains the ID of a user and their gender.
The table has an equal number of 'female', 'male', and 'other'.
```

Write an SQL query to rearrange the `Genders` table such that the rows alternate between 'female', 'other', and 'male' in order. The table should be rearranged such that the IDs of each gender are sorted in ascending order.

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

The query result format is shown in the following example.

Example 1:**

```
Input:
Genders table:
+---------+--------+
| user_id | gender |
+---------+--------+
| 4       | male   |
| 7       | female |
| 2       | other  |
| 5       | male   |
| 3       | female |
| 8       | male   |
| 6       | other  |
| 1       | other  |
| 9       | female |
+---------+--------+
Output:
+---------+--------+
| user_id | gender |
+---------+--------+
| 3       | female |
| 1       | other  |
```

```
| 4        | male   |
| 7        | female |
| 2        | other  |
| 5        | male   |
| 9        | female |
| 6        | other  |
| 8        | male   |
+---------+--------+
```
Explanation:
Female gender: IDs 3, 7, and 9.
Other gender: IDs 1, 2, and 6.
Male gender: IDs 4, 5, and 8.
We arrange the table alternating between 'female', 'other', and 'male&#3
Note that the IDs of each gender are sorted in ascending order.