2408. Design SQL

Description

You are given `n` tables represented with two arrays `names` and `columns`, where `names[i]` is the name of the `i th` table and `columns[i]` is the number of columns of the `i th` table.

You should be able to perform the following **operations**:

- **Insert** a row in a specific table. Each row you insert has an id. The id is assigned using an auto-increment method where the id of the first inserted row is 1, and the id of each other row inserted into the same table is the id of the last inserted row (even if it was deleted) plus one.
- **Delete** a row from a specific table. **Note** that deleting a row does not affect the id of the next inserted row.
- **Select** a specific cell from any table and return its value.

Implement the `SQL` class:

- `SQL(String[] names, int[] columns)` Creates the `n` tables.
- `void insertRow(String name, String[] row)` Adds a row to the table `name`. It is **guaranteed** that the table will exist, and the size of the array `row` is equal to the number of columns in the table.
- `void deleteRow(String name, int rowId)` Removes the row `rowId` from the table `name`. It is **guaranteed** that the table and row will **exist**.
- `String selectCell(String name, int rowId, int columnId)` Returns the value of the cell in the row `rowId` and the column `columnId` from the table `name`.

Example 1:

```
Input
["SQL", "insertRow", "selectCell", "insertRow", "deleteRow", "selectCell"]
[[["one", "two", "three"], [2, 3, 1]], ["two", ["first", "second", "third"]], ["two", 1, 3], ["two", ["fourth", "fifth", "sixth"]], ["two", 1], ["two", 2, 2]]
Output
[null, null, "third", null, null, "fifth"]

Explanation
SQL sql = new SQL(["one", "two", "three"], [2, 3, 1]); // creates three tables.
sql.insertRow("two", ["first", "second", "third"]); // adds a row to the table "two". Its id is 1.
sql.selectCell("two", 1, 3); // return "third", finds the value of the third column in the row with id 1 of the table "two".
sql.insertRow("two", ["fourth", "fifth", "sixth"]); // adds another row to the table "two". Its id is 2.
sql.deleteRow("two", 1); // deletes the first row of the table "two". Note that the second row will still have the id 2.
sql.selectCell("two", 2, 2); // return "fifth", finds the value of the second column in the row with id 2 of the table "two".
```

Constraints:

- `n == names.length == columns.length`
- `1 <= n <= 10 4`
- `1 <= names[i].length, row[i].length, name.length <= 20`
- `names[i]`, `row[i]`, and `name` consist of lowercase English letters.
- `1 <= columns[i] <= 100`
- All the strings of `names` are **distinct**.
- `name` exists in the array `names`.
- `row.length` equals the number of columns in the chosen table.
- `rowId` and `columnId` will be valid.
- At most `250` calls will be made to `insertRow` and `deleteRow`.
- At most `10 4` calls will be made to `selectCell`.