



DEVELOPER

S

[Getting Started](#)[Examples](#)Link records with duplicate values

On this page

Link records with duplicate values

! INFO

Select a record and the column to be checked, the script will automatically find the rest of the duplicate records and then associate them to the selected record (you need to create a Two-way link field in advance)

Demo

New datasheetManager

Grid viewNew view

Insert recordHide fieldsFilterGroupSortRow heightShareFindFormMirrorAPIAdvanced

	Title	Options	Check
1	Test1		
2	Test1	One	
3	Test1	Four	
4	APITable.com		
5	xukecheng@apitable.com		
6	Test1	One	
7	Test1	Two	
8	Test1	Three	
9	+1-650-555-1234		
10	+1-800-555-1212		
11	+44-800-123-4567		
12	+81-3-1234-5678		
13	+972-3-6123456		
14	+61-2-1234-5678		
15	+33-1-23-45-67-89		
16	+49-89-12345678		
17	+31-20-123-4567		
18	+55-11-2345-6789		
19	12345-67890-1		
20	09876-54321-2		
21	98765-43210-3		
22	56789-01234-4		
23	45678-90123-5		

28 records

Widget board

Script

Edit codeRun

Please select the field to store duplicate records, the type needs to be magic link and the association is the current datasheet:

Please select an option

Console

Source code

```

const datasheet = await space.getActiveDatasheetAsync();

const storeField = await input.fieldAsync(
  "Please select the field to store duplicate records, the type needs to be two-way link and the association is the
  datasheet
);
if (storeField.type !== "TwoWayLink") {
  output.text(
    "Failed to run: The selected field type needs to be a Two-way link"
  );
} else if (storeField.property.foreignDatasheetId !== datasheet.id) {
  output.text(
    "Failed to run: the selected magical link field must be linked to current datasheet"
  );
} else {
  console.log(storeField.property);

  const needFindRecord = await input.recordAsync(
    "Please select the record to check:",
    datasheet
  );
  const needFindField = await input.fieldAsync(
    "Please select the fields to check for duplicates:",
    datasheet
  );
};

const records = await datasheet.getRecordsAsync();

// Define two data sets that need to be used later, duplicates stores the duplicate record ids found, and valuesM
const duplicates = [];
const valuesMap = {};

for (let record of records) {
  //The data that is empty does not belong to the content to be found, so it is skipped
  if (record.getCellValueString(needFindField.id) === null) {
    continue;
  }
  //Need to exclude yourself when checking for duplicates
  if (record.id === needFindRecord.id) {
    continue;
  }
  //Obtain data according to getCellValueString and judge whether they are equal. If they are equal, it means tha
  if (
    record.getCellValueString(needFindField.id) ===
    needFindRecord.getCellValueString(needFindField.id)
  ) {
    duplicates.push(record.id);
  }
}

console.log(duplicates);

if (duplicates.length === 0) {
  output.text(`No duplicate records found`);
} else {
  output.text(
    `Found ${duplicates.length} duplicate records, which have been linked in the Two-way link field`
  );
  valuesMap[storeField.id] = duplicates;
  await datasheet.updateRecordAsync(needFindRecord.id, valuesMap);
}
}

```

[Previous](#)

[« Find the phone number in a column](#)

[Next](#)

[Extract URL from attachment »](#)

Developer Center

[Development guide](#)

[API Reference](#)

Social

[Twitter](#) 

More

[Homepage](#) 

[Help Center](#) 

[GitHub](#) 