Survey Designer and surveyResponse JSON

Last Modified on 12/14/2025 8:45 pm CST

Understanding SurveyJSON | Understanding ResponseJSON | Working with ResponseJSON in SQL Server

The latest iteration of the Survey Designer tool, originally released in Campus.2040, is built using a JavaScript library called SurveyJS. Full documentation of the SurveyJS library is available. This latest version of the tool saves data in a different way than the previous tool, so any user-created SQL scripts that point to survey response tables will need to be adjusted. Both the survey and the survey responses are now stored as a string of JavaScript Object Notation (JSON). JSON supports hierarchical data structures, making use of objects and arrays.

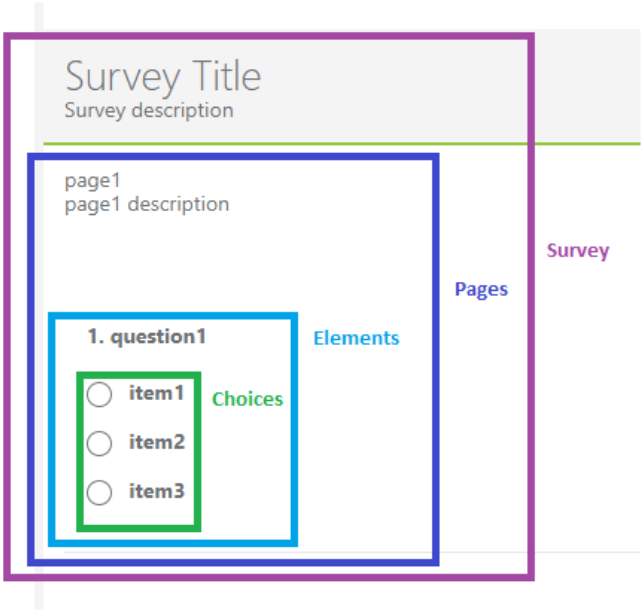
An object is a key/value pair. (e.q., {"key1" : "value1", "key2":"value2"})

An array is a list of values. e.q., [value1, value2, value3]

Within the JSON, objects and arrays can contain other objects or arrays, giving the data a *tree-like* structure. eq., {"key1" : [{ "values1" : [1, 2], "values2" : [3,4] }] }

Understanding SurveyJSON

When building a survey in the Survey Designer, the user begins with an empty survey and is able to add components like pages, and questions. The survey components translate to Objects in the SurveyJSON string.



#	Survey Component	JSON Object	contains things like...
1	Survey Body	ROOT object	Title, description, survey properties, containers
2	Containers	Pages	page title, page description, page properties, elements
3	Questions	Elements	question properties, Choices

In the surveyJSON, Survey is the Root Object that contains the pages and other survey properties like title and description. Pages are the top-level container object within the survey. Pages can contain elements. Elements can contain question properties, and question choices. Choices can contain Items.

For example, the following surveyJSON describes a survey with two pages, each with page title and page description filled in, and with one question on each page:

{"title":"Survey Title","description":"Survey description","**pages**":[{"name":"*page1*","**elements**":[{"type":"radiogroup","name":"question1","**choices**":["item1","item2","item3"]}],"title":"page1","description":"page1 description"},{"name":"*page2*","**elements**":[{"type":"checkbox","name":"question2","title":"question2","**choices**":["item1","item2","item3"]}],"title":"page2","description":"page2 description"}]}

Formatting the JSON makes it easier to understand the object hierarchy (google JSON Pretty Print to find an online formatting tool like https://jsonformatter.org/json-pretty-print).

SurveyJSON Formatted


```
{
  "title": "Survey Title",
  "description": "Survey description",
  "pages": [
    {
      "name": "page1",
      "elements": [
        {
          "type": "radiogroup",
          "name": "question1",
          "choices": [
            "item1",
            "item2",
            "item3"
          ]
        }
      ],
      "title": "page1",
      "description": "page1 description"
    },
    {
      "name": "page2",
      "elements": [
        {
          "type": "checkbox",
          "name": "question2",
          "title": "question2",
          "choices": [
            "item1",
            "item2",
            "item3"
          ]
        }
      ],
      "title": "page2",
      "description": "page2 description"
    }
  ]
}
```

Understanding ResponseJSON

Compared to the SurveyJSON, the ResponseJSON is a bit simpler to understand. When a user responds to a survey, the JSON string containing that user's complete response is saved to the surveyResponse table in the responseJSON column. The responseJSON stores the respondent's name and question answers. If the survey was anonymous, the respondent name is not stored.

For example, the following responseJSON is what would be saved if user Carlota Johnson selected item1 in question 1, and then item2 and item3 in question 2 in the example survey from the previous section:

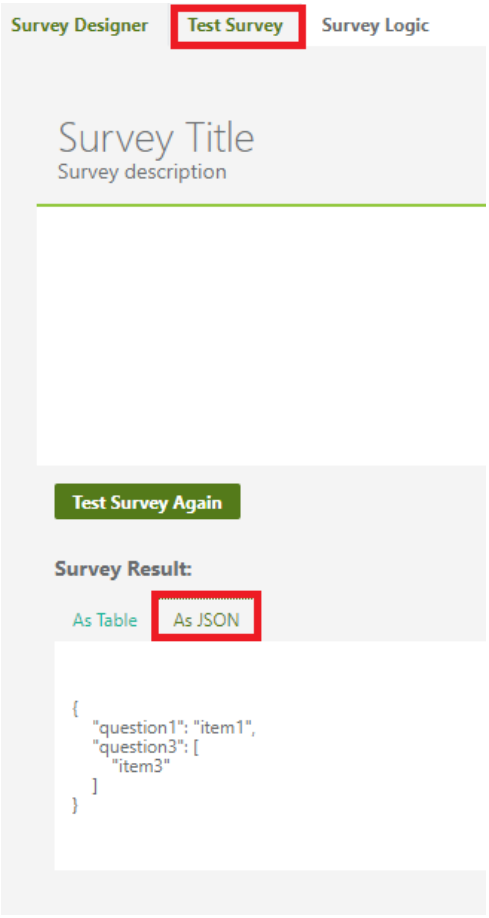
{"firstname":"CARLOTA","question1":"item1","question2":["item2","item3"],"lastname":"JOHNSON"}

ResponseJSON Formatted


```
{
  "firstname": "CARLOTA",
  "question1": "item1",
  "question2": [
    "item2",
    "item3"
  ],
  "lastname": "JOHNSON"
}
```


> The name in the ResponseJSON is static. That means if the user's name changes in the future, the name within the survey will not change. This functionality might change in the future.

The ResponseJSON can be viewed when testing the survey.



Working with ResponseJSON in SQL Server

Working with JSON in Microsoft SQL Server is possible. To see in-depth SQL examples, we recommend reviewing the official Microsoft documentation for the **OPENJSON** function:

https://docs.microsoft.com/en-us/sql/t-sql/functions/openjson-transact-sql?view=sql-server-ver15

A simple SQL OPENJSON Example

This SQL statement will return data from the surveyResponse table along with the key : value pairs in the responseJSON based on a specified responseID.

1. Declare a variable for responseID
2. Set the variable
3. Select the desired columns from surveyResponse table along with the key and value from the responseJSON
4. Cross join the ResonseJSON results with the surveyResponse results where responseID is the same

```
--declare a variable for responseID
DECLARE @responseID INT
SET @responseID = 11

--select the desired columns from surveyResponse table along with the key and value from the responseJSON
SELECT surveyID
    ,personID
    ,respondentID
    ,startTimestamp
    ,endTimestamp
    ,rj.[key] AS question
    ,rj.value AS answer
FROM surveyResponse sr
CROSS APPLY OPENJSON((  --cross join the ResonseJSON results with the surveyResponse results where responseID is the same
            SELECT responseJSON
            FROM surveyResponse
            WHERE responseID = @responseID
            )) AS rj
WHERE responseID = @responseID
```

Example Result set

surveyID	personID	respondentID	startTimestamp	endTimestamp	question	answer
1	1234	1	2021-04-22 13:23:09.307	2021-04-22 13:23:15.730	firstname	CARLOTA
1	1234	1	2021-04-22 13:23:09.307	2021-04-22 13:23:15.730	question2	["item2","item3"]
1	1234	1	2021-04-22 13:23:09.307	2021-04-22 13:23:15.730	question1	item1
1	1234	1	2021-04-22 13:23:09.307	2021-04-22 13:23:15.730	lastname	JOHNSON