<https://blogs.technet.microsoft.com/dataplatforminsider/2016/01/08/json-in-sql-server-2016-part-4-of-4/>

# JSON in SQL Server 2016: Part 4 of 4

★★★★★

★★★★

★★★

★★

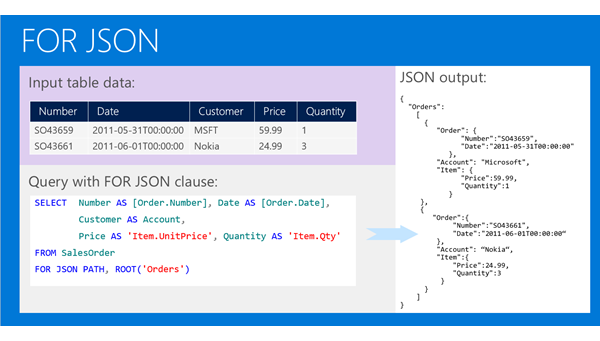
★

January 8, 2016 by [Jovan Popovic (MSFT)](https://blogs.technet.microsoft.com/dataplatforminsider/author/jovan-popovic-msft/) // [1 Comments](https://blogs.technet.microsoft.com/dataplatforminsider/2016/01/08/json-in-sql-server-2016-part-4-of-4/#comments)

* [2](https://blogs.technet.microsoft.com/dataplatforminsider/2016/01/08/json-in-sql-server-2016-part-4-of-4/)
* [3](https://blogs.technet.microsoft.com/dataplatforminsider/2016/01/08/json-in-sql-server-2016-part-4-of-4/)

# Exporting Data as JSON – FOR JSON

In this final post of our four-part JSON series, we showcase the ability to format query results as JSON text using the FOR JSON clause. If you are familiar with the FOR XML clause, you will easily understand FOR JSON.

[](https://msdnshared.blob.core.windows.net/media/TNBlogsFS/prod.evol.blogs.technet.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/00/60/54/FOR%20JSON%20clause.png)

When you add the FOR JSON clause at the end of a SQL SELECT query, SQL Server will take the results, format them as JSON text, and return them to the client. Every row will be formatted as one JSON object, values in cells of the result set will be generated as values of JSON objects, and column names or aliases will be used as key names. We have two kinds of FOR JSON clauses:

* FOR JSON PATH: Enables you to define the structure of an output JSON using the column names/aliases. If you put dot-separated names in the column aliases, JSON properties will follow the naming convention. This feature is similar to FOR XML PATH where you can use slash separated paths.
* FOR JSON AUTO Automatically creates nested JSON sub arrays based on the table hierarchy used in the query. Again, this is similar to FOR XML AUTO.

JSON text that is generated with a FOR JSON clause can be transformed back to the relational form using OPENJSON.

# Conclusion

JSON functions in SQL Server enable you to query and analyze JSON data as well as transform JSON to relational domains, and relational data to JSON. They allow you to integrate SQL Server with external systems that produce or consume JSON data without additional transformations in the application layer.

SQL Server also provides a hybrid storage model where you can combine relational data and JSON. This model enables you to make trade-offs between high performance data access and flexibility/rapid developments. You can use the same indexing techniques both on standard columns and values in JSON text.

The hybrid model retains all the benefits of the SQL Server engine with fully powerful query language and ACID transactions. It also boasts well-known management and security models, several features that work with JSON functions, and a wide ecosystem of tools compatible with SQL Server.

Check out the other posts in this four-part series in the links below, or learn more in the [SQL Server 2016 blogging series](https://blogs.technet.microsoft.com/b/dataplatforminsider/archive/2015/12/01/get-the-most-out-of-sql-server-2016.aspx?wt.mc_id=WW_CE_DM_OO_BLOG_NONE).

JSON in SQL Server 2016: [Part 1 of 4](https://blogs.technet.microsoft.com/b/dataplatforminsider/archive/2016/01/05/json-in-sql-server-2016-part-1-of-4.aspx?wt.mc_id=WW_CE_DM_OO_BLOG_NONE)

JSON in SQL Server 2016: [Part 2 of 4](https://blogs.technet.microsoft.com/b/dataplatforminsider/archive/2016/01/06/json-in-sql-server-2016-part-2-of-4.aspx?wt.mc_id=WW_CE_DM_OO_BLOG_NONE)

JSON in SQL Server 2016: [Part 3 of 4](https://blogs.technet.microsoft.com/b/dataplatforminsider/archive/2016/01/07/json-in-sql-server-2016-part-3-of-4.aspx?wt.mc_id=WW_CE_DM_OO_BLOG_NONE)