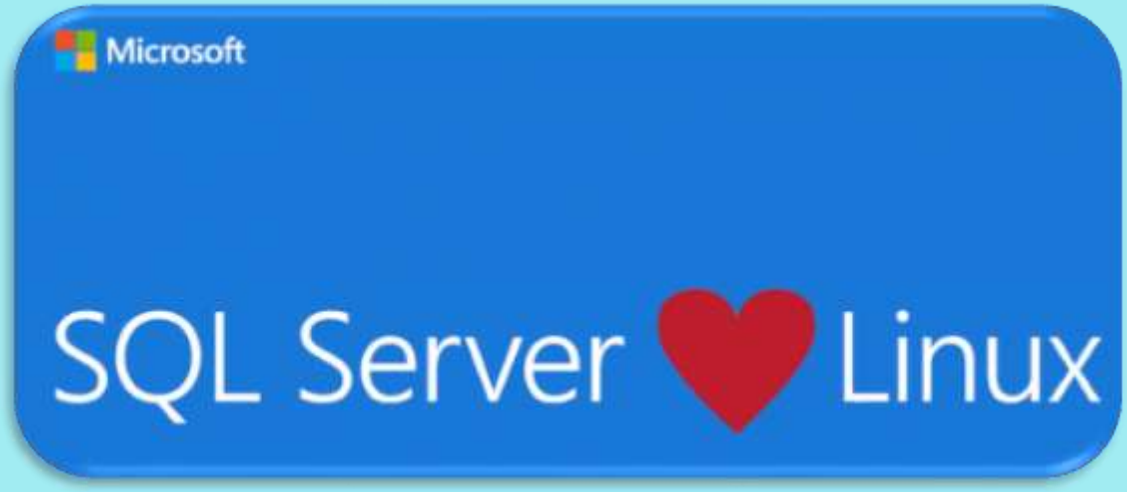
豐聯資訊

講者：David

時間：2018/06/05

目錄

跨出 Windows 的第一個版本



原先只支援 Windows 作業系統的狀況，變成在 Linux、Docker 以及微軟的雲端環境 Azure 也可以執行

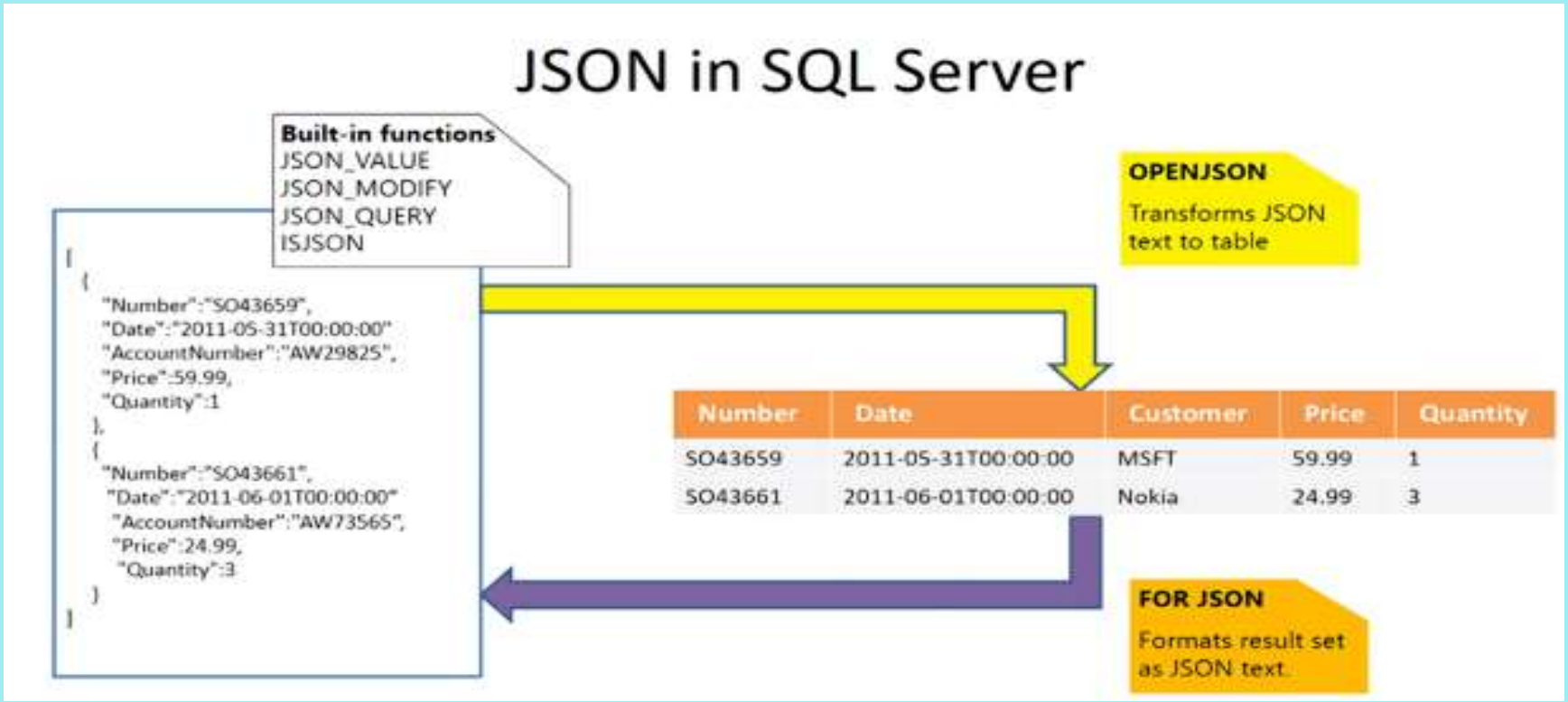
SQLServer
如何接收JSON字串並解析

接收資料類型使用NVARCHAR。

NVARCHAR(max) 可儲存 2 GB 的 JSON 文件。

若確定 JSON 文件不大於 8 KB，基於效能考量，建議使用 NVARCHAR(4000) 而不要使用 NVARCHAR(max)。

SQLServer
如何接收JSON字串並解析



SQLServer
如何接收JSON字串並解析

OPENJSON：將JSON文字轉換成一組資料列和資料行

JSON_VALUE：從 JSON 字串擷取**純量值**

JSON_MODIFY：變更 JSON 字串中的值

JSON_QUERY：從 JSON 字串**擷取物件或陣列**

ISJSON：測試字串是否為有效的 JSON

SQLServer
如何接收JSON字串並解析

1. OPENJSON：可將JSON文字轉換成一或多個資料列

選項 1：
預設輸出的 OPENJSON，此函式會傳回包含下列三個資料行的資料表

1. 輸入**物件**中的**屬性名稱** (或輸入**陣列**中元素的**索引**)。
2. 屬性或陣列元素的**值**。
3. **類型** (例如字串、數字、布林值、陣列或物件)。

SQLServer
如何接收JSON字串並解析

豐聯資訊

範例：JSON物件

```
-- 選項1 具有預設輸出的OPENJSON
DECLARE @json NVARCHAR(4000)

--object
SET @json='{"string":"John",
            "number":45,
            "boolean":true,
            "array":["SQL","C#","MVC"],
            "object": { "objKey": "objVal"}
           }'

SELECT *
  FROM OPENJSON(@json)
```

SQLServer
如何接收JSON字串並解析

結果

Key	Value	Type
string	John	1
number	45	2
boolean	True	3
array	["SQL","C#","MVC"]	4
object	{"objKey":"objVal"}	5

SQLServer
如何接收JSON字串並解析

範例：JSON陣列

```
--array
SET @json='[{"string":"John",
            "number":45,
            "boolean":true,
            "array":["SQL","C#","MVC"],
            "object": { "objKey": "objVal"}
          }]'

SELECT *
  FROM OPENJSON(@json)
```

SQLServer
如何接收JSON字串並解析

結果

Key	Value	Type
0	{"string": "John", "number": 45, "boolean": true, "array":["SQL","C#","MVC"], "object": {"objKey": "objVal"} }	5

SQLServer如何接收JSON字串並解析

選項 2：
具有明確結構的OPENJSON輸出，只會傳回在**WITH子句中**所**定義之資料行的資料表**。

選擇性 **WITH** 子句中，可以**指定**一組**輸出資料行**、**類型**，以及每個**輸出值的 JSON 來源屬性路徑**。

SQLServer
如何接收JSON字串並解析

豐聯資訊

範例：JSON陣列

```
SET @json =
  N'[
      {
        "Data": { "Id":"B123456789",
                  "Number":"S111111",
                  "Date":"2011/05/31",
                  "Quantity":10
                },
        "ActionFlag":"I"
      },
      {
        "Data": {
          "Id":"B123456789",
          "Number":"S222222",
          "Date":"2011/05/30",
          "Quantity":20
        },
        "ActionFlag":"U"
     }]'

SELECT * FROM OPENJSON ( @json )
WITH (          Number      varchar(200)  '$.Data.Number' ,
                Date        datetime      '$.Data.Date',
                ActionFlag  varchar(1)    '$.ActionFlag',
                Quantity    int           '$.Data.Quantity'
     )
```

SQLServer
如何接收JSON字串並解析

豐聯資訊

結果

Number	Date	ActionFlag	Quantity
S111111	2011/05/31	I	10
S222222	2011/05/30	U	20
S333333	2011/05/29	D	30

SQLServer
如何接收JSON字串並解析

JSON_VALUE ：語法 → JSON_VALUE（ expression , path ）

引數 1(expression)	引數 2(path)
包含 JSON 文字的變數的名稱	要擷取之屬性的 JSON 路徑

SQLServer
如何接收JSON字串並解析

豐聯資訊

小測驗

```
DECLARE @jsonInfo NVARCHAR(4000)

SET @jsonInfo=N'{
     "info":{
       "type":1,
       "address":{
         "town":"Bristol",
         "county":"Avon",
         "country":"England"
       },
       "tags":["Sport", "Water polo"]
    },
    "type":"Basic"
 }'

SELECT JSON_VALUE(@jsonInfo, '$'),JSON_VALUE(@jsonInfo, '$.info.type'),
       JSON_VALUE(@jsonInfo, '$.info."address"'),JSON_VALUE(@jsonInfo, '$.info.tags'),
       JSON_VALUE(@jsonInfo, '$.info.type[0]'), JSON_VALUE(@jsonInfo, '$.info.none'),
       JSON_VALUE(@jsonInfo, '$.info.address.town')
```

SQLServer
如何接收JSON字串並解析

豐聯資訊

路徑	值	備註
$	Null	非純量值，改用 **JSON_QUERY**
$info.type	1	
$info.address.town	Bristol	
$info."address"	Null	非純量值，改用 **JSON_QUERY**
$info.tags	Null	非純量值，改用 **JSON_QUERY**
$info.type[0]	Null	非陣列
$info. none	Null	屬性不存在

SQLServer
如何接收JSON字串並解析

JSON_QUERY ：語法 → JSON_QUERY(expression , path)

引數 1(expression)	引數 2(path)
包含 JSON 文字的變數的名稱	要擷取之屬性的 JSON 路徑

SQLServer
如何接收JSON字串並解析

豐聯資訊

小測驗

```
SET @jsonInfo=N'{
    "info": {
        "type": 1,
        "address": {
            "town": "Bristol",
            "county": "Avon",
            "country": "England"
        },
        "tags": ["Sport", "Water polo"]
    },
    "type": "Basic"
   } '

SELECT JSON_QUERY(@jsonInfo, '$'),                JSON_QUERY(@jsonInfo, '$.info.type'),
       JSON_QUERY(@jsonInfo, '$.info."address"'), JSON_QUERY(@jsonInfo, '$.info.tags'),
       JSON_QUERY(@jsonInfo, '$.info.type[0]'),   JSON_QUERY(@jsonInfo, '$.info.none'),
       JSON_QUERY(@jsonInfo, '$.info.address.town')
```

SQLServer
如何接收JSON字串並解析

豐聯資訊

路徑	值	備註
$	回傳完整JSON文字	
$info.type	Null	非物件/陣列，改用 **JSON_VALUE**
$info.address.town	Null	非物件/陣列，改用 **JSON_VALUE**
$info."address"	{ "town":"Bristol", "county":"Avon", "country":"England" }	
$info.tags	["Sport", "Water polo"]	
$info.type[0]	Null	非陣列
$info. none	Null	屬性不存在

SQLServer
如何接收JSON字串並解析

JSON_MODIFY：
語法 → JSON_MODIFY（ expression, path, newValue ）

引數 1(expression)	引數 2(path)	引數 3(newValue)
含有 JSON 文字的變數的名稱	指定更新之屬性JSON路徑 *若前有append修飾詞，則指定新的值附加到<JSON路徑>參考的陣列	path指定之屬性的新值

SQLServer
如何接收JSON字串並解析

範例

ActionScript	Output
SET @info	{"name":"John","skills":["C#","SQL"]}
SET @info=JSON_MODIFY(@info,'$.name','Mike')	{ "name": "Mike", "skills": ["C#", "SQL"] }
SET @info=JSON_MODIFY(@info,'$.surname','Smith')	{ "name": "Mike", "skills": ["C#", "SQL"], "surname": "Smith" }
SET @info=JSON_MODIFY(@info,'$.name',NULL)	{ "skills": ["C#", "SQL"], "surname": "Smith" }
SET @info=JSON_MODIFY(@info,'append $.skills','Azure')	{ "skills": ["C#", "SQL", "Azure"], "surname": "Smith" }

SQLServer
如何接收JSON字串並解析

多個更新：一個JSON_MODIFY，更新一個屬性；多個更新，可以使用多個 JSON_MODIFY 呼叫。

```
--Multiple Update
DECLARE @info NVARCHAR(100)='{"name":"John","skills":["C#","SQL"]}'

SET @info=JSON_MODIFY(JSON_MODIFY(JSON_MODIFY(@info,'$.name','Mike'),'$.surname','Smith'),'append $.skills','Azure')

PRINT @info
{"name":"Mike","skills":["C#","SQL","Azure"],"surname":"Smith"}
```

SQLServer
如何接收JSON字串並解析

豐聯資訊

ISJSON：語法 → ISJSON（ expression ）

引數(expression)	傳回值
要測試的字串	包含**有效**的 JSON，則**傳回 1**；**否則傳回 0** 若 expression 為 null，則傳回 null 不會傳回錯誤

SQLServer
如何接收JSON字串並解析

豐聯資訊

範例

```
DECLARE @jsonInfo NVARCHAR(4000)

--SET @jsonInfo=N'{  ...
SET @jsonInfo=N'{
    "info": {
        "type": 1,
        "address": {
            "town": "Bristol",
            "county": "Avon",
            "country": "England"
        },
        "tags": ["Sport", "Water polo"]
    },
    "type": "Basic"
  } '


SELECT (CASE WHEN (ISJSON(@jsonInfo) > 0) THEN 'Y' ELSE 'N' END)
```

常見問題

1. SQLServer版本在2016以上就支援JSON資料，但輸入相關JSON語法會有問題。

```
DECLARE @json NVARCHAR(4000)
SET @json = N'{
 "info":{
     "type":1,
     "address":{
         "town":"Bristol",
         "county":"Avon",
         "country":"England"
     },
     "tags":["Sport", "Water polo"]
 },
 "type":"Basic"
}'

SELECT * FROM OPENJSON(@json, N'lax $.info')
```

121 %

訊息

訊息 208，層級 16，狀態 1，行 15
無效的物件名稱 'OPENJSON'。

解決辦法：
ALTER DATABASE database_name
SET COMPATIBILITY_LEVEL = 130

常見問題

SQLServer 版本	相容性層級指定
SQL Server 2017	140
SQL Server 2016	130
SQL Server 2014	120
SQL Server 2012	110

可以用下列語法範例檢視database的相容性層級↓

```
select compatibility_level from sys.databases
where name ='PMDB'
```

參考文獻

MSDN：https://goo.gl/K55RHr、https://goo.gl/an5c7s
https://goo.gl/ZhbFr1、 https://goo.gl/SgYVkV

SQL Database Engine Blog：https://goo.gl/dvpkH7

補充說明

Q：若DB本身屬性為SQL Server 2016以下，即相容性層級為130以下，更改相容性層級後可以使用OPENJSON功能嗎?

A：無法使用，以範例來看，本機上為2012的版本，若用SQL Server 2017開啟，更改相容性層級會發生錯誤，意即版本只能向下相容，不能向上相容。

補充說明

範例：

```
select compatibility_level from sys.databases
where name ='Work'
```

100 %

結果 訊息

	compatibility_level
1	110

```
ALTER DATABASE Work SET COMPATIBILITY_LEVEL = 130
```

100 %

訊息

訊息 15048，層級 16，狀態 1，行 8
資料庫相容性層級的有效值為 90、100 或 110。

補充說明

Q：傳進來的JSON陣列，是否可以識別DATETIME格式?

A：可以，但需要事先知道是屬於DATETIME2 還是 DATETIME格式。

若預設值型態DATETIME2 但使用DATETIME接收會發生錯誤，不過若預設型態為DATETIME 但使用DATETIME2接收**不會發生錯誤**

補充說明

範例：

```
SET @I_CHR_JSON =N'[{"DateTime2":"2018-06-06 16:15:08.9093310","DateTime":"2018-06-08 09:37:53.897"}]'

SELECT * FROM OPENJSON(@I_CHR_JSON)
WITH( DateTime2 DATETIME2 '$.DateTime2',
      DateTime  DATETIME  '$.DateTime'
    )
```

100 %

結果 訊息

	DateTime2	DateTime
1	2018-06-06 16:15:08.9093310	2018-06-08 09:37:53.897

```
SET @I_CHR_JSON =N'[{"DateTime2":"2018-06-06 16:15:08.9093310","DateTime":"2018-06-08 09:37:53.897"}]'

SELECT * FROM OPENJSON(@I_CHR_JSON)
WITH( DateTime2 DATETIME2 '$.DateTime2',
      DateTime  DATETIME2 '$.DateTime'
    )
```

100 %

結果 訊息

	DateTime2	DateTime
1	2018-06-06 16:15:08.9093310	2018-06-08 09:37:53.8970000

補充說明

若預設型態為DATETIME2，接收型態為DATETIME則會出錯

```
SET @I_CHR_JSON =N'[{"DateTime2":"2018-06-06 16:15:08.9093310","DateTime":"2018-06-08 09:37:53.897"}]'

SELECT * FROM OPENJSON(@I_CHR_JSON)
WITH( DateTime2 DATETIME  '$.DateTime2',
      DateTime  DATETIME2 '$.DateTime'
    )
```

100 %

結果 訊息

訊息 241，層級 16，狀態 1，行 6
從字元字串轉換成日期及/或時間時，轉換失敗。

補充說明

若傳進來為JSON物件，使用OPENJSON打開則會視為字串型態，若不加上引號(“”)包覆則會出錯

```
SET @I_CHR_JSON =N'{"DateTime2":"2018-06-06 16:15:08.9093310","DateTime":"2018-06-08 09:37:53.897"}'

SELECT * FROM OPENJSON(@I_CHR_JSON)
```

100 %

結果 訊息

	key	value	type
1	DateTime2	2018-06-06 16:15:08.9093310	1
2	DateTime	2018-06-08 09:37:53.897	1

補充說明

以下範例為JSON陣列↓

```
[
    {
        "ActionFlag": "I",
        "Data": {
            "CORPID": "ILS", "BRNCD": "TC", "EMPNO": "1060501",
            "SENIORITY": 1, "NOTE": {"TITLE": "Programmer","AGE": 20},
            "DATASOURCE": [
                { "value": "1", "label": "养老长期照护入住", "key": "1" },
                { "value": "2", "label": "喘息、短托入住", "key": "2"},
                { "value": "3", "label": "其他原因入住", "key": "3" }
            ]
        }
    },
    {
        "ActionFlag": "U",
        "Data": {
            "CORPID": "ILS", "BRNCD": "TC", "EMPNO": "1000501",
            "SENIORITY": 5, "NOTE": { "TITLE": "SA", "AGE": 28 },
            "DATASOURCE": [
                { "value": "1", "label": "特级", "key": "1" },
                { "value": "2", "label": "一级", "key": "2" },
                { "value": "3", "label": "二級", "key": "3" },
                { "value": "4", "label": "三級", "key": "4" }
            ]
        }
    }
]
```

補充說明

先從存取第一層屬性開始，會發現明明就有Data這個屬性，為什麼會存取會是NULL? Why?????

```
SELECT * FROM OPENJSON(@I_CHR_JSON)
  WITH ( ACTIONFLAG VARCHAR(1)      '$.ActionFlag',
         Data       NVARCHAR(4000) '$.Data')
 WHERE ISJSON(@I_CHR_JSON) > 0
```

100 %

結果 訊息

	ACTIONFLAG	Data
1	I	NULL
2	U	NULL

補充說明

資料行定義中使用AS JSON指定參考屬性包含JSON物件或陣列，同時資料行的型態必須為NVARCHAR(MAX)，這點和JSON_QUERY的含式行為類似

```
SELECT * FROM OPENJSON(@I_CHR_JSON)
  WITH ( ACTIONFLAG VARCHAR(1)      '$.ActionFlag',
         Data       NVARCHAR(MAX)   '$.Data' AS JSON)
 WHERE ISJSON(@I_CHR_JSON) > 0
```

100 %

結果 訊息

	ACTIONFLAG	Data
1	I	{"CORPID":"ILS","BRNCD":"TC","EMPNO":"1060501","...
2	U	{"CORPID":"ILS","BRNCD":"TC","EMPNO":"1000501","...

補充說明

開始存取第二層，結果如下↓

```
SELECT * FROM OPENJSON(@I_CHR_JSON)
 WITH ( ACTIONFLAG VARCHAR(1)    '$.ActionFlag',
        Data       NVARCHAR(MAX) '$.Data'         AS JSON,
        CORPID     VARCHAR(10)   '$.Data.CORPID',
        BRNCD      VARCHAR(5)    '$.Data.BRNCD',
        EMPNO      VARCHAR(7)    '$.Data.EMPNO',
        SENIORITY  INT           '$.Data.SENIORITY',
        NOTE       NVARCHAR(MAX) '$.Data.NOTE'     AS JSON,
        DATASOURCE NVARCHAR(MAX) '$.Data.DATASOURCE' AS JSON) AS ROOT
```

100 %

結果 訊息

	ACTIONFLAG	Data	CORPID	BRNCD	EMPNO	SENIORITY	NOTE	DATASOURCE
1	I	{"CORPID":"ILS",...	ILS	TC	1060501	1	{"TITLE":"Programmer","AGE":20}	[{"value":"1","label":"养老长期照护入住
2	U	{"CORPID":"ILS",...	ILS	TC	1000501	5	{"TITLE":"SA","AGE":28}	[{"value":"1","label":"特级","key":"1"},

註：非純量值，就要用AS JSON

補充說明

Q：如何存取NOTE屬性的JSON物件資料?

A：利用第一層的WITH子句(輸出資料行)，加上CROSS APPLY這個含式，即可將NOTE屬性底下的Key(TITLE、AGE) 取出

```
SELECT NOTE.TITLE, NOTE.AGE FROM OPENJSON(@I_CHR_JSON)
 WITH ( ACTIONFLAG VARCHAR(1)     '$.ActionFlag',
        Data       NVARCHAR(MAX) '$.Data'          AS JSON,
        CORPID     VARCHAR(10)   '$.Data.CORPID',
        BRNCD      VARCHAR(5)    '$.Data.BRNCD',
        EMPNO      VARCHAR(7)    '$.Data.EMPNO',
        SENIORITY  INT           '$.Data.SENIORITY',
        NOTE       NVARCHAR(MAX) '$.Data.NOTE'     AS JSON,
        DATASOURCE NVARCHAR(MAX) '$.Data.DATASOURCE' AS JSON) AS ROOT
 CROSS APPLY OPENJSON(ROOT.NOTE)
  WITH(
        TITLE VARCHAR(20),
        AGE   INT
      ) AS NOTE
```

100 %

結果 訊息

	TITLE	AGE
1	Programmer	20
2	SA	28

補充說明

Q：如何存取DATASOURCE屬性的JSON陣列資料?

A：解法一樣，利用第一層的WITH子句輸出的資料結構，加上CROSS APPLY這個函式，即可將DATASOURCE屬性底下的JSON陣列取出，記得要加上路徑(' $.keyValue')，範例在下一個頁面

補充說明

範例：

```
SELECT DATASOURCE.VALUE, DATASOURCE.LABEL, DATASOURCE.[KEY] FROM OPENJSON(@I_CHR_JSON)
 WITH ( ACTIONFLAG VARCHAR(1)     '$.ActionFlag',
        Data       NVARCHAR(MAX) '$.Data'          AS JSON,
        CORPID     VARCHAR(10)   '$.Data.CORPID',
        BRNCD      VARCHAR(5)    '$.Data.BRNCD',
        EMPNO      VARCHAR(7)    '$.Data.EMPNO',
        SENIORITY  INT           '$.Data.SENIORITY',
        NOTE       NVARCHAR(MAX) '$.Data.NOTE'     AS JSON,
        DATASOURCE NVARCHAR(MAX) '$.Data.DATASOURCE' AS JSON) AS ROOT
 CROSS APPLY OPENJSON(ROOT.DATASOURCE)
  WITH(
       VALUE VARCHAR(1)   '$.value',
       LABEL NVARCHAR(20) '$.label',
       [KEY] VARCHAR(1)   '$.key'
      ) AS DATASOURCE
 WHERE ISJSON(@I_CHR_JSON) > 0
```

100 %

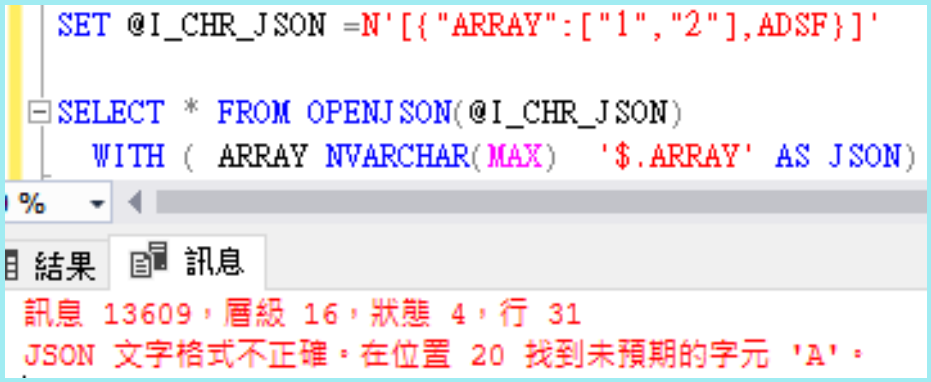
結果 訊息

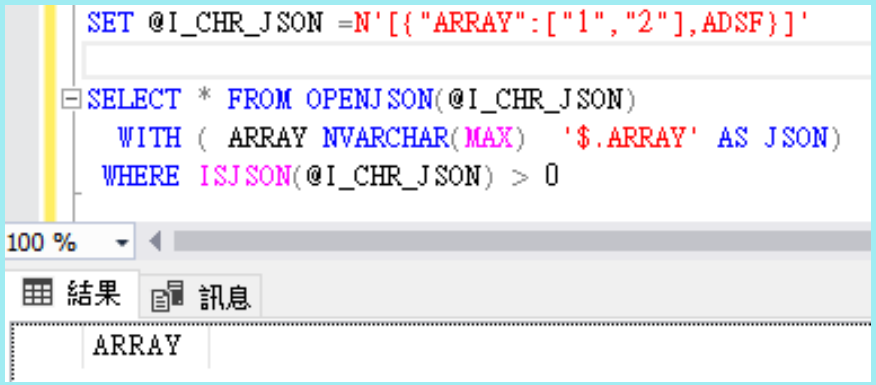
	VALUE	LABEL	KEY
1	1	养老长期照护入住	1
2	2	喘息、短托入住	2
3	3	其他原因入住	3
4	1	特级	1
5	2	一级	2
6	3	二級	3
7	4	三級	4

補充說明

有可能傳進來的JSON字串格式是有誤的，直接使用會拋出格式錯誤訊息，如下方左圖。

為了避免錯誤發生，可以使用ISJSON函式防止出錯，如右圖





THE END