bite-sized.sql

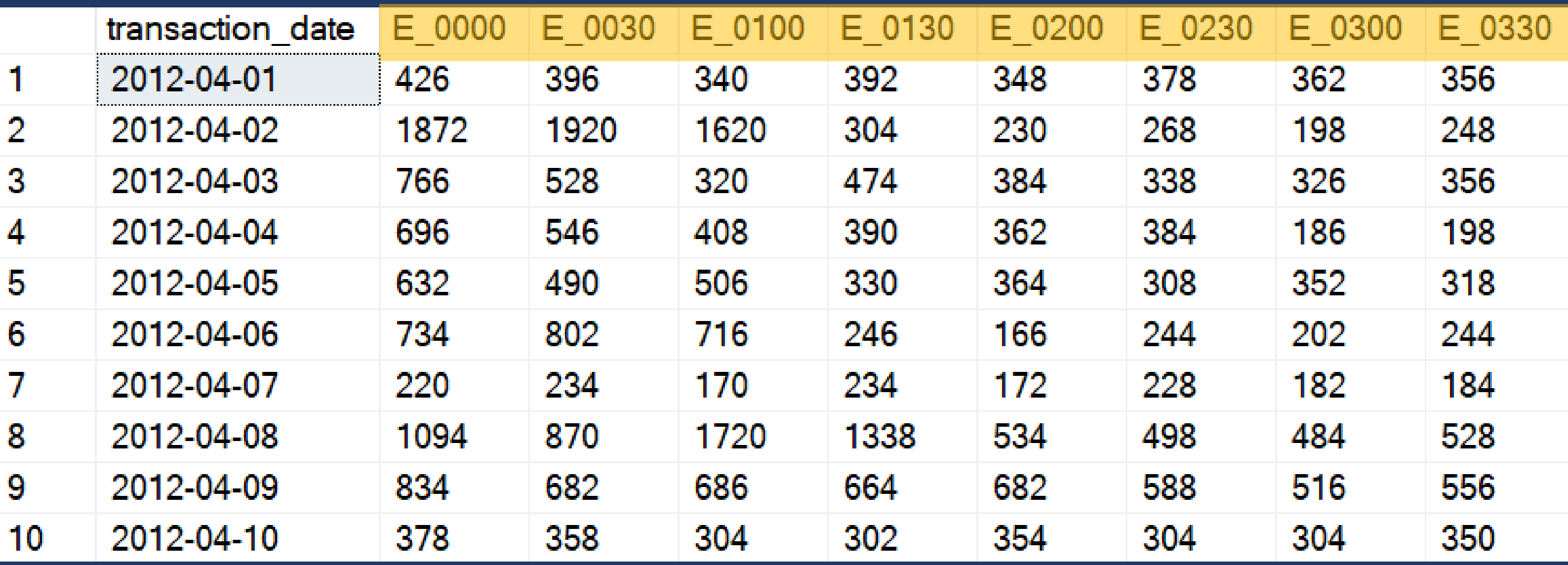
SQL: UNPIVOT columns with JSON

What's the problem?

Suppose you have a table with data encoded in the column headers

You want to turn this:

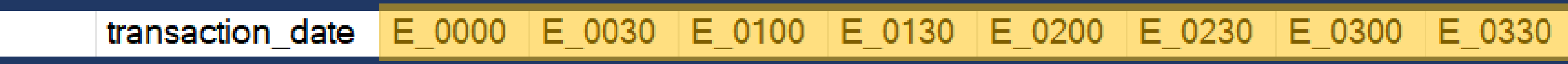
See those codes? They mean something (half hour time slots). That's data. Data encoded as column headers is problematic.



	transaction_date	E_0000	E_0030	E_0100	E_0130	E_0200	E_0230	E_0300	E_0330
1	2012-04-01	426	396	340	392	348	378	362	356
2	2012-04-02	1872	1920	1620	304	230	268	198	248
3	2012-04-03	766	528	320	474	384	338	326	356
4	2012-04-04	696	546	408	390	362	384	186	198
5	2012-04-05	632	490	506	330	364	308	352	318
6	2012-04-06	734	802	716	246	166	244	202	244
7	2012-04-07	220	234	170	234	172	228	182	184
8	2012-04-08	1094	870	1720	1338	534	498	484	528
9	2012-04-09	834	682	686	664	682	588	516	556
10	2012-04-10	378	358	304	302	354	304	304	350

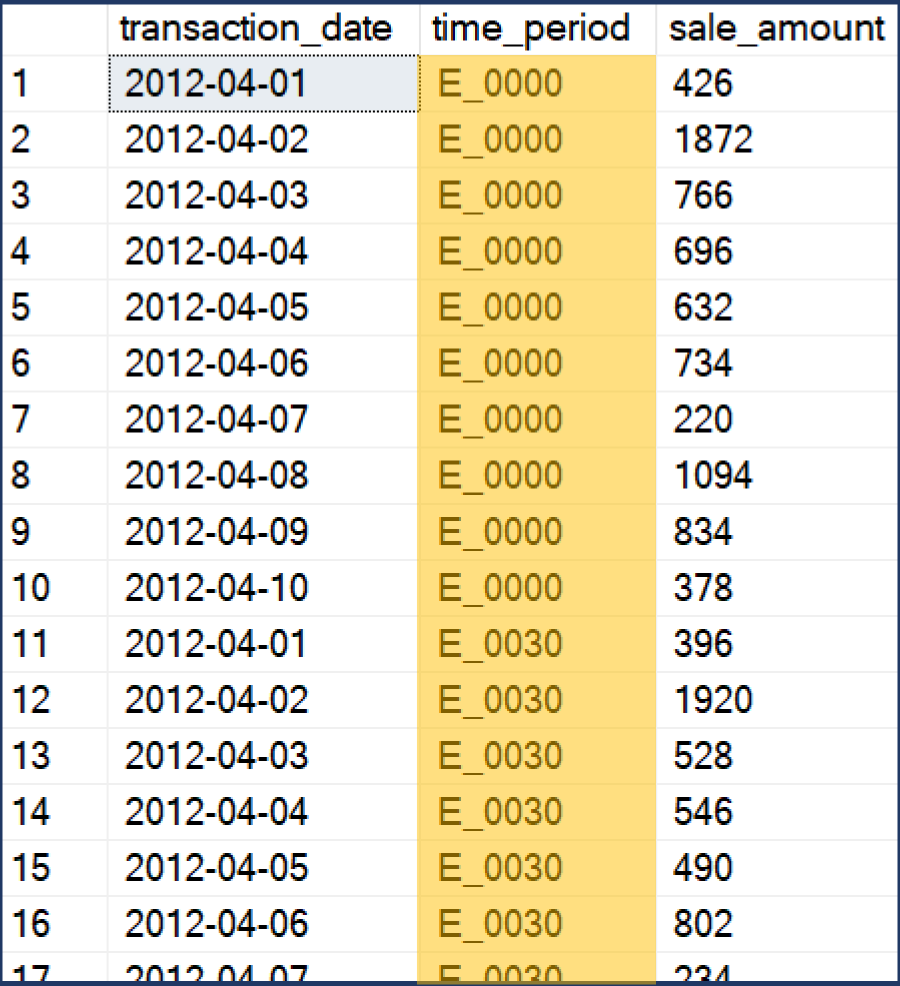
In a relational database, this format will be much more useful for querying

We need to *UNPIVOT* every column except a specific column



transaction_date	E_0000	E_0030	E_0100	E_0130	E_0200	E_0230	E_0300	E_0330

Into this:



	transaction_date	time_period	sale_amount
1	2012-04-01	E_0000	426
2	2012-04-02	E_0000	1872
3	2012-04-03	E_0000	766
4	2012-04-04	E_0000	696
5	2012-04-05	E_0000	632
6	2012-04-06	E_0000	734
7	2012-04-07	E_0000	220
8	2012-04-08	E_0000	1094
9	2012-04-09	E_0000	834
10	2012-04-10	E_0000	378
11	2012-04-01	E_0030	396
12	2012-04-02	E_0030	1920
13	2012-04-03	E_0030	528
14	2012-04-04	E_0030	546
15	2012-04-05	E_0030	490
16	2012-04-06	E_0030	802
17	2012-04-07	E_0030	234

We can do this with UNPIVOT, but this requires us to type out the values of the column we’re unpivoting

```sql
SELECT
    transaction_date,
    time_period,
    sale_amount
FROM
    input1
UNPIVOT (
    sale_amount FOR time_period IN (
        E_0000, E_0030, E_0100, E_0130,
        E_0200, E_0230, E_0300, E_0330
    )
) AS unpvt;
```

The column headers of the columns to be unpivoted must be explicitly listed!

This might be a problem if we have lots of columns, or if we don’t know how many columns there are at runtime

We can get round this using dynamic SQL and INFORMATION_SCHEMA to get the column names

```
DECLARE @columns NVARCHAR(MAX);
DECLARE @sql NVARCHAR(MAX);

SELECT @columns = STRING_AGG(QUOTENAME(COLUMN_NAME),', ')
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'input1'
AND COLUMN_NAME <> 'transaction_date';

SET @sql = '
SELECT transaction_date, time_period, sale_amount
FROM input1
UNPIVOT
(sale_amount
FOR time_period IN (' + @columns + ')) AS unpvt;';

EXEC sp_executesql @sql;
```

QUOTENAME wraps a text value in square brackets

3 Execute the query

2 Concatenate the CSV of column names into the query as a dynamic SQL string

1 Get a CSV of the column names we want to unpivot from INFORMATION_SCHEMA.COLUMNS.

All that just to unpivot some columns?!

```
DECLARE @                    (MAX);
DECLARE                          ;

SELECT @colu          ING_AGG(QUOTENAME(COLUMN_NAME),', ')
FROM INFORMA           MA.COLUMNS
WHERE TABLE_           nput1'
AND COLUMN_             ansaction_date';

SET @sql =
SELECT tra              , time_period, sale_amount
FROM input
UNPIVOT
(sale_amo
FOR time_               @columns + ')) AS unpvt;';

EXEC sp_exe                ;
```

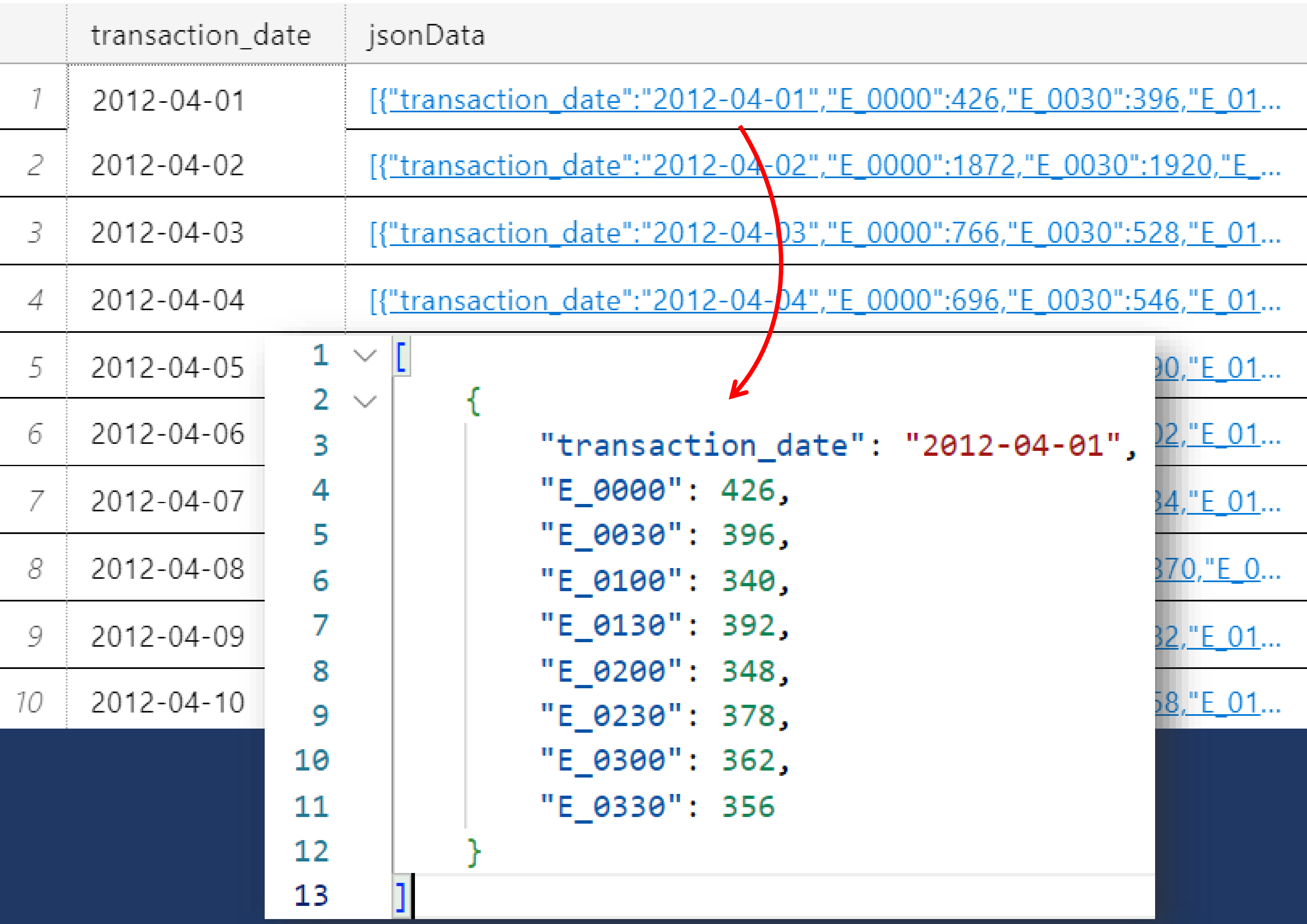
Thankfully there’s another way!

flexyourdata.com | youtube.com/@flexyourdata | linkedin.com/in/owenhprice

JSON to the rescue!

First let's note that FOR JSON PATH will convert a record into a JSON array

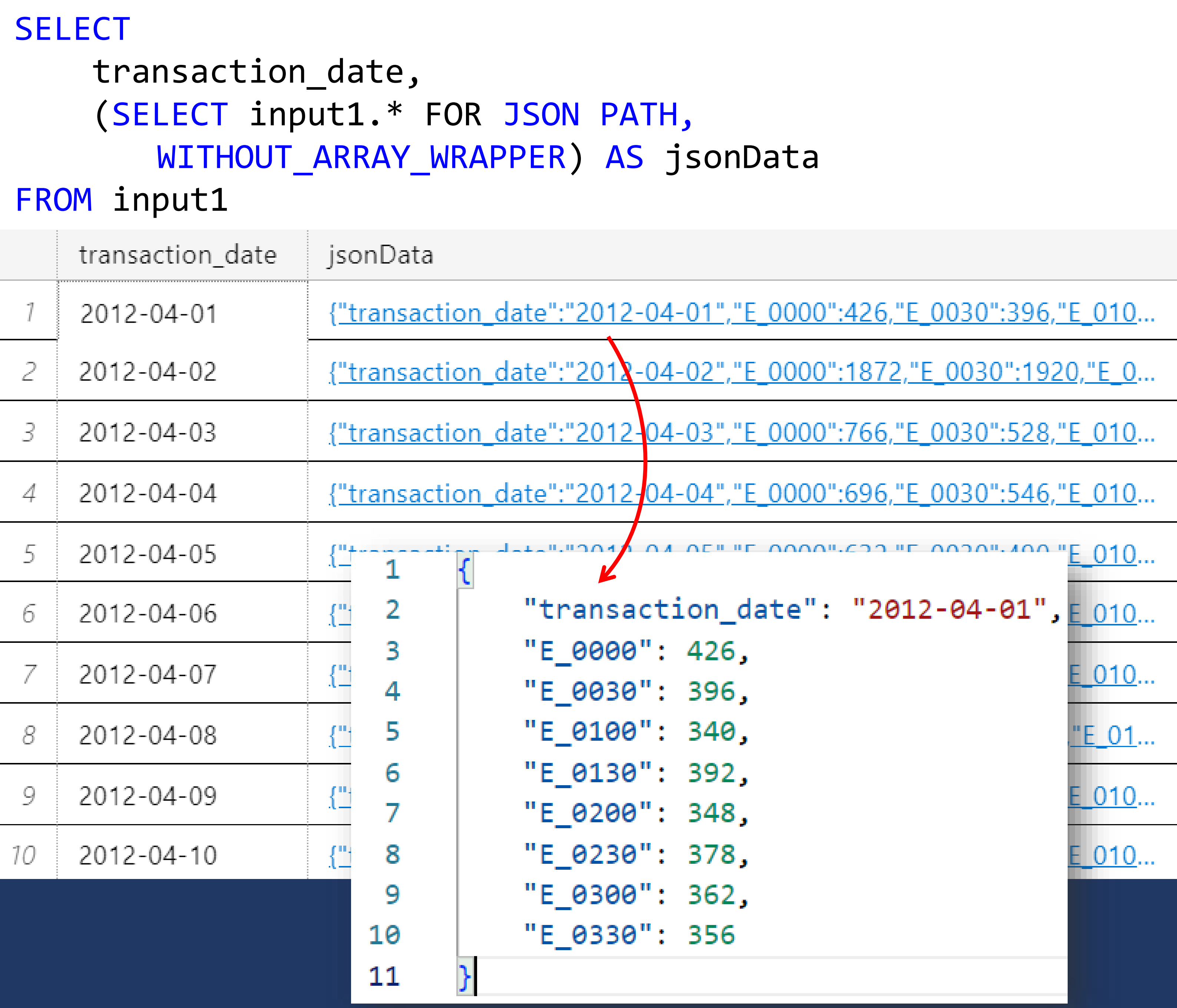
```sql
SELECT
    transaction_date,
    (SELECT input1.* FOR JSON PATH) AS jsonData
FROM input1
```



	transaction_date	jsonData
1	2012-04-01	[{"transaction_date":"2012-04-01","E_0000":426,"E_0030":396,"E_01...
2	2012-04-02	[{"transaction_date":"2012-04-02","E_0000":1872,"E_0030":1920,"E_...
3	2012-04-03	[{"transaction_date":"2012-04-03","E_0000":766,"E_0030":528,"E_01...
4	2012-04-04	[{"transaction_date":"2012-04-04","E_0000":696,"E_0030":546,"E_01...
5	2012-04-05	90,"E_01...
6	2012-04-06	02,"E_01...
7	2012-04-07	34,"E_01...
8	2012-04-08	370,"E_0...
9	2012-04-09	32,"E_01...
10	2012-04-10	58,"E_01...

```json
[
    {
        "transaction_date": "2012-04-01",
        "E_0000": 426,
        "E_0030": 396,
        "E_0100": 340,
        "E_0130": 392,
        "E_0200": 348,
        "E_0230": 378,
        "E_0300": 362,
        "E_0330": 356
    }
]
```

If we add WITHOUT_ARRAY_WRAPPER, it becomes a JSON object



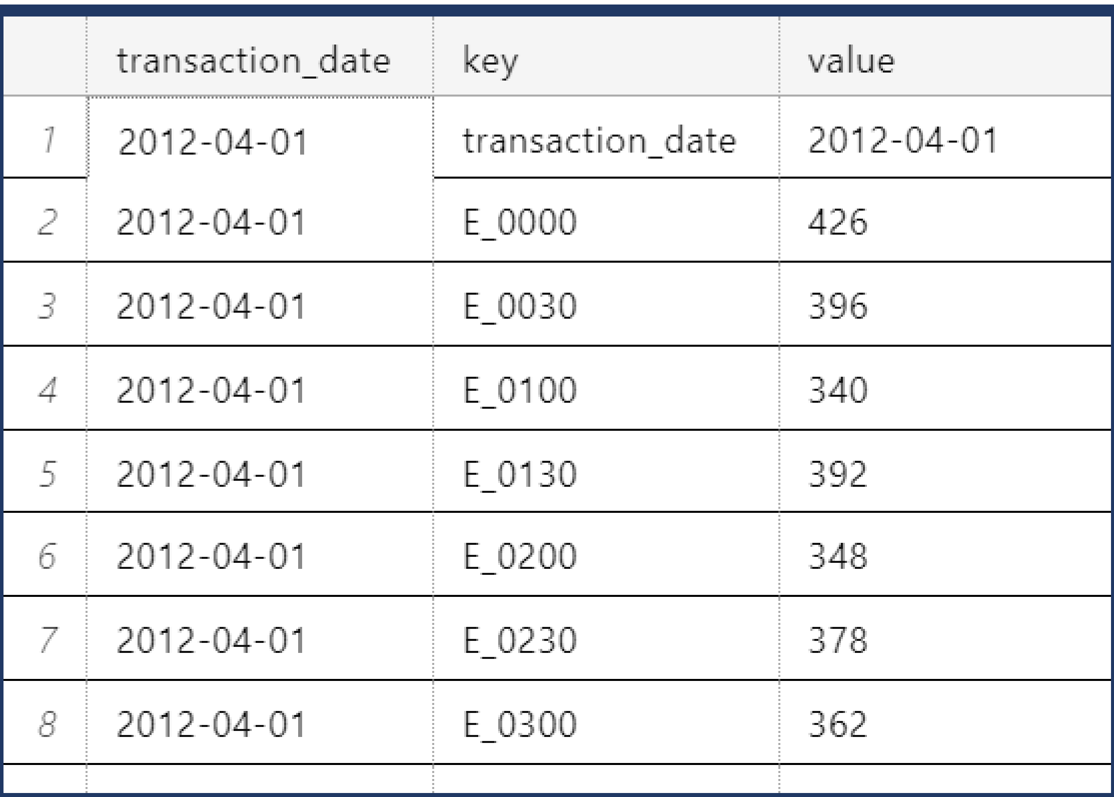
```sql
SELECT
    transaction_date,
    (SELECT input1.* FOR JSON PATH,
       WITHOUT_ARRAY_WRAPPER) AS jsonData
FROM input1
```

	transaction_date	jsonData
1	2012-04-01	{"transaction_date":"2012-04-01","E_0000":426,"E_0030":396,"E_010...
2	2012-04-02	{"transaction_date":"2012-04-02","E_0000":1872,"E_0030":1920,"E_0...
3	2012-04-03	{"transaction_date":"2012-04-03","E_0000":766,"E_0030":528,"E_010...
4	2012-04-04	{"transaction_date":"2012-04-04","E_0000":696,"E_0030":546,"E_010...
5	2012-04-05	{"transaction_date":"2012-04-05","E_0000":632,"E_0030":490,"E_010...
6	2012-04-06	{" E_010...
7	2012-04-07	{" E_010...
8	2012-04-08	{" "E_01...
9	2012-04-09	{" E_010...
10	2012-04-10	{" E_010...

```json
{
    "transaction_date": "2012-04-01",
    "E_0000": 426,
    "E_0030": 396,
    "E_0100": 340,
    "E_0130": 392,
    "E_0200": 348,
    "E_0230": 378,
    "E_0300": 362,
    "E_0330": 356
}
```

We can extract values from JSON using the OPENJSON function

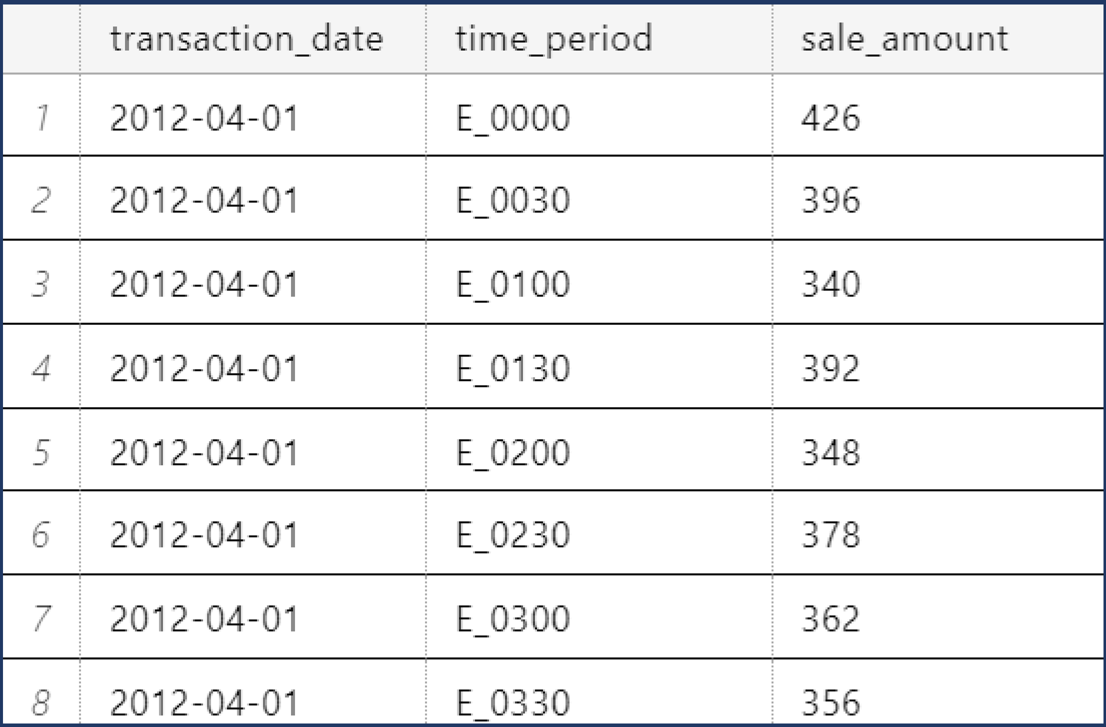
```sql
WITH JSONData AS (
    SELECT
        transaction_date,
        (SELECT input1.* FOR JSON PATH,
        WITHOUT_ARRAY_WRAPPER) AS jsonData
    FROM input1
)
SELECT
    transaction_date,
    j.[key],
    j.[value]
FROM JSONData
CROSS APPLY OPENJSON(jsonData) AS j
```



	transaction_date	key	value
1	2012-04-01	transaction_date	2012-04-01
2	2012-04-01	E_0000	426
3	2012-04-01	E_0030	396
4	2012-04-01	E_0100	340
5	2012-04-01	E_0130	392
6	2012-04-01	E_0200	348
7	2012-04-01	E_0230	378
8	2012-04-01	E_0300	362

All we need to do is filter out the transaction date and alias the columns!

```
WITH JSONData AS (
    SELECT
        transaction_date,
        (SELECT input1.* FOR JSON PATH,
        WITHOUT_ARRAY_WRAPPER) AS jsonData
    FROM input1
)
SELECT
    transaction_date,
    j.[key] AS time_period,
    j.[value] AS sale_amount
FROM JSONData
CROSS APPLY OPENJSON(jsonData) AS j
WHERE j.[key] <> 'transaction_date'
```



	transaction_date	time_period	sale_amount
1	2012-04-01	E_0000	426
2	2012-04-01	E_0030	396
3	2012-04-01	E_0100	340
4	2012-04-01	E_0130	392
5	2012-04-01	E_0200	348
6	2012-04-01	E_0230	378
7	2012-04-01	E_0300	362
8	2012-04-01	E_0330	356

Takeaways

1. The UNPIVOT keyword requires a list of the columns to be unpivoted
2. We can build this list using dynamic SQL and the INFORMATION_SCHEMA.COLUMNS table
3. We can convert a row to a JSON array using FOR JSON PATH
4. The WITHOUT_ARRAY_WRAPPER option will return a JSON object instead of a JSON array
5. OPENJSON allows us to extract the keys and values from a JSON object