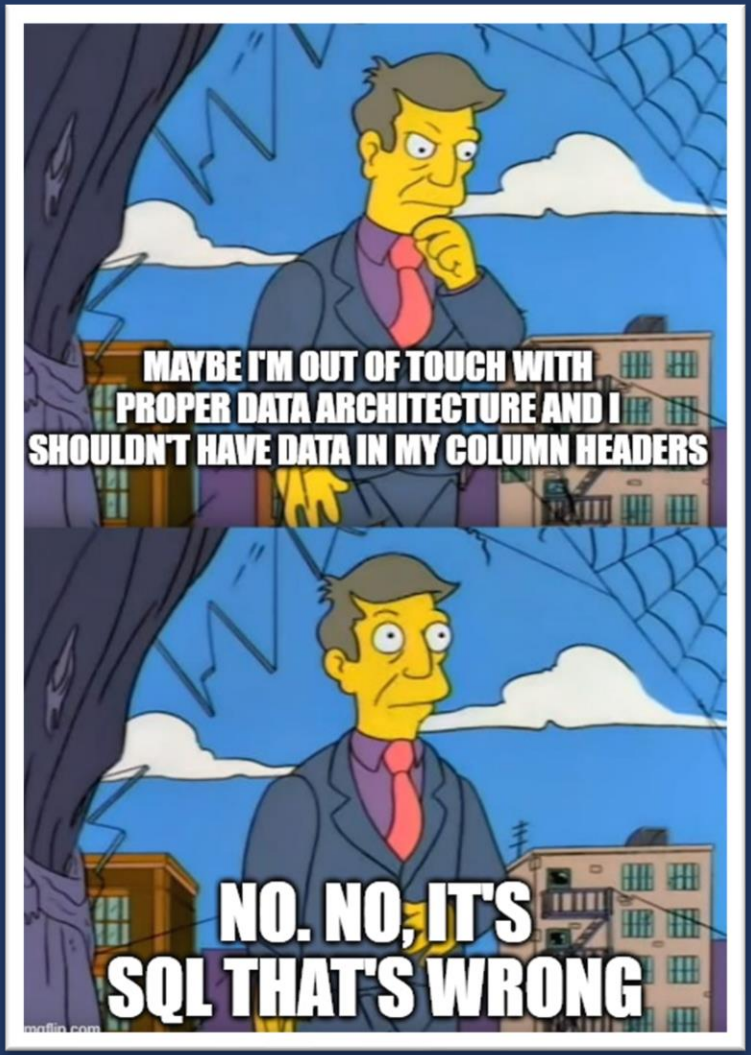
SQL: DYNAMIC UNPIVOT OF COLUMNS



Suppose we have this source data

```
SELECT *
FROM SOURCE;
```

	Item	202110	202111	202112	202201	202202	202203
1	A	1	2	4	2	3	4
2	B	2	4	8	4	5	6
3	C	3	6	12	6	7	8
4	D	4	8	16	8	9	10
5	E	5	10	20	10	11	12
6	A	NULL	NULL	NULL	NULL	4	12
7	D	NULL	NULL	NULL	NULL	8	1
8	E	NULL	NULL	NULL	NULL	12	21

And we want to normalize the columns into {attribute, value} pairs

From this

	Item	202110	202111	202112	202201	202202	202203
1	A	1	2	4	2	3	4
2	B	2	4	8			
3	C	3	6	12			
4	D	4	8	16			
5	E	5	10	20			
6	A	NULL	NULL	NULL			
7	D	NULL	NULL	NULL			
8	E	NULL	NULL	NULL			

To this

	Item	YearMonth	RawValue
1	A	202110	1
2	A	202111	2
3	A	202112	4
4	A	202201	2
5	A	202202	3
6	A	202203	4
7	B	202110	2
8	B	202111	4
9	B	202112	8
10	B	202201	4

UNPIVOT of columns

The problem with this approach is we need to *hard-code* the column names in the IN

We list the columns we want to see in the output

```
SELECT Item, YearMonth, RawValue
FROM (
    SELECT * FROM SOURCE
) p
UNPIVOT
    (RawValue
    FOR YearMonth
        IN (
            [202110], [202111], [202112],
            [202201], [202202], [202203]
            )
    ) unpvt
```

SELECT from a sub-query that returns the data we want to unpivot

We then use the UNPIVOT keyword, followed by a parenthesized clause

The text in UNPIVOT takes the form
X FOR Y IN (list of columns)
Here, X is the output column where we want to put the values from the columns being unpivoted and Y represents the column names of the columns being unpivoted.

Dynamic UNPIVOT of columns:

(i) build the column name string

STRING_AGG concatenates the values from different rows into a single value with an optional row-separator

QUOTENAME prepares a string as a valid SQL Server object identifier by wrapping it in square brackets

```
SELECT STRING_AGG(QUOTENAME(COLUMN_NAME), ', ') as cols
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'source'
AND COLUMN_NAME <> 'Item’
```

We can SELECT the columns names from any table using INFORMATION_SCHEMA.COLUMNS

	cols
1	[202110], [202111], [202112], [202201], [202202], [202203]

Dynamic UNPIVOT of columns:

(ii) concatenate and insert to #temp table

Use nvarchar(4000) variable when creating dynamic SQL in SQL Server

Create a temp table which will hold the unpivoted data

```
DECLARE @sql nvarchar(4000);

DROP TABLE IF EXISTS #unpivot;
CREATE TABLE #unpivot (Item varchar(1), YearMonth varchar(6), RawValue int);

SELECT @sql =
'
INSERT INTO #unpivot (Item, YearMonth, RawValue)
SELECT Item, YearMonth, RawValue
FROM (
    SELECT * FROM SOURCE
) p
UNPIVOT
(RawValue FOR YearMonth IN (' + cols + ')) unpvt'
FROM (
    SELECT STRING_AGG(QUOTENAME(COLUMN_NAME), ', ') as cols
    FROM INFORMATION_SCHEMA.COLUMNS
    WHERE TABLE_NAME = 'source'
    AND COLUMN_NAME <> 'Item'
) c;

EXEC(@sql);
```

Concatenate the result of the INFORMATION_SCHEMA query with the UNPIVOT query, and SELECT the result into the @sql variable

Executing the dynamic SQL inserts the unpivoted data into the temp table

Dynamic UNPIVOT of columns:

(iii) the result

```
SELECT * FROM #unpivot;
```

The result is the unpivoted data is achieved *without* hard-coding column names.

	Item	YearMonth	RawValue
1	A	202110	1
2	A	202111	2
3	A	202112	4
4	A	202201	2
5	A	202202	3
6	A	202203	4
7	B	202110	2
8	B	202111	4
9	B	202112	8
10	B	202201	4
11	B	202202	5
12	B	202203	6
13	C	202110	3
14	C	202111	6

Takeaways

1. Use UNPIVOT to normalize cross-tabulated attributes

2. Use STRING_AGG and QUOTENAME on a query of INFORMATION_SCHEMA.COLUMNS to dynamically build the IN list in UNPIVOT

3. Concatenate the dynamic column list with the UNPIVOT query and insert the result into a temp table