


Use a Format File to Skip a Table Column (SQL Server)

Article • 04/04/2023

Applies to:  [SQL Server](#)  [Azure SQL Database](#)  [Azure SQL Managed Instance](#)  [Azure Synapse Analytics](#)  [Analytics Platform System \(PDW\)](#)

This article describes how to use a format file to skip importing a table column when the data for the skipped column does not exist in the source data file. A data file can contain fewer fields than the number of columns in the destination table - that is, you can skip importing a column - only if at least one of the following two conditions is true in the destination table:

- The skipped column is nullable.
- The skipped column has a default value.

Note

This syntax, including bulk insert, is not supported in Azure Synapse Analytics. In Azure Synapse Analytics and other cloud database platform integrations, accomplish data movement via **the COPY statement in Azure Data Factory**, or by using **T-SQL statements such as COPY INTO and PolyBase**.

Sample table and data file

The examples in this article expect a table named `myTestSkipCol` under the `dbo` schema. You can create this table in a sample database such as `WideWorldImporters` or `AdventureWorks` or in any other database. Create this table as follows:

SQL

```
USE WideWorldImporters;
GO
CREATE TABLE myTestSkipCol
(
    Col1 smallint,
    Col2 nvarchar(50) NULL,
    Col3 nvarchar(50) not NULL
);
GO
```

The examples in this article also use a sample data file, `myTestSkipCol2.dat`. This data file contains only two fields, although the destination table contains three columns.

Output

```
1,DataForColumn3
1,DataForColumn3
1,DataForColumn3
```

Basic steps

You can use a non-XML format file or an XML format file to skip a table column. In both cases, there are two steps:

1. Use the [bcp command-line utility](#) to create a default format file.
2. Modify the default format file in a text editor.

The modified format file must map each existing field to its corresponding column in the destination table. It must also indicate which table column or columns to skip.

For example, to bulk import data from `myTestSkipCol2.dat` into the `myTestSkipCol` table, the format file must map the first data field to `col1`, skip `col2`, and map the second field to `col3`.

Option #1 - Use a non-XML format file

Step #1 - Create a default non-XML format file

Create a default non-XML format file for the `myTestSkipCol` sample table by running the following **bcp** command at the command prompt:

Windows Command Prompt

```
bcp WideWorldImporters..myTestSkipCol format nul -f
myTestSkipCol_Default.fmt -c -T
```

Important

You might have to specify the name of the server instance to which you are connecting with the `-s` argument. Also, you might have to specify the user

name and password with the `-U` and `-P` arguments. For more information, see [bcp Utility](#).

The previous command creates a non-XML format file, `myTestSkipCol_Default.fmt`. This format file is called a *default format file* because it is the form generated by **bcp**. A default format file describes a one-to-one correspondence between data-file fields and table columns.

The following screenshot shows the values in this sample default format file.

| | Version | Number of columns | Host file data type | Host file data length | Prefix length | Terminator | Server column order | Server column name | Column collation |
|--|---------|-------------------|---------------------|-----------------------|---------------|------------|---------------------|--------------------|------------------------------|
| | 9.0 | 3 | SQLCHAR | 0 | 7 | "\t" | 1 | Col1 | "" |
| | | | SQLCHAR | 0 | 100 | "\t" | 2 | Col2 | SQL_Latin1_General_CP1_CI_AS |
| | | | SQLCHAR | 0 | 100 | "\r\n" | 3 | Col3 | SQL_Latin1_General_CP1_CI_AS |

Note

For more information about the format-file fields, see [non-XML format files \(SQL Server\)](#).

Step #2 - Modify a non-XML format file

To modify a default non-XML format file, there are two alternatives. Either alternative indicates that the data field does not exist in the data file and that no data is to be inserted into the corresponding table column.

To skip a table column, edit the default non-XML format file and modify the file by using one of the following alternative methods:

Option #1 - Remove the row

The preferred method for skipping a column involves the following three steps:

1. First, delete any format-file row that describes a field that is missing from the source data file.
2. Then, reduce the "Host file field order" value of each format-file row that

- follows a deleted row. The goal is sequential "Host file field order" values, 1 through n , that reflect the actual position of each data field in the data file.
3. Finally, reduce the value in the "Number of columns" field to reflect the actual number of fields in the data file.

The following example is based on the default format file for the `myTestSkipCol` table. This modified format file maps the first data field to `col1`, skips `col2`, and maps the second data field to `col3`. The row for `col2` has been deleted. The delimiter after the first field has also been changed from `\t` to `,`.

| Output | | | | | | | |
|------------------------------|---------|---|-----|--------|---|------|----|
| 14.0 | | | | | | | |
| 2 | | | | | | | |
| 1 | SQLCHAR | 0 | 7 | "," | 1 | Col1 | "" |
| 2 | SQLCHAR | 0 | 100 | "\r\n" | 3 | Col3 | |
| SQL_Latin1_General_CP1_CI_AS | | | | | | | |

Option #2 - Modify the row definition

Alternatively, to skip a table column, you can modify the definition of the format-file row that corresponds to the table column. In this format-file row, the "prefix length," "host file data length," and "server column order" values must be set to 0. Also, the "terminator" and "column collation" fields must be set to "" (that is, to an empty or NULL value). The "server column name" value requires a non-blank string, though the actual column name is not necessary. The remaining format fields require their default values.

The following example is also derived from the default format file for the `myTestSkipCol` table.

| Output | | | | | | | |
|------------------------------|---------|---|-----|--------|---|------|----|
| 14.0 | | | | | | | |
| 3 | | | | | | | |
| 1 | SQLCHAR | 0 | 7 | "," | 1 | Col1 | "" |
| 2 | SQLCHAR | 0 | 0 | "" | 0 | Col2 | "" |
| 3 | SQLCHAR | 0 | 100 | "\r\n" | 3 | Col3 | |
| SQL_Latin1_General_CP1_CI_AS | | | | | | | |

Examples with a non-XML format file

The following examples are based on the `myTestSkipCol` sample table and the `myTestSkipCol2.dat` sample data file that are described earlier in this article.

Use BULK INSERT

This example works by using either of the modified non-XML format files created as described in the preceding section. In this example, the modified format file is named `myTestSkipCol2.fmt`. To use `BULK INSERT` to bulk import the `myTestSkipCol2.dat` data file, in SQL Server Management Studio (SSMS), run the following code. Update the file system paths for the location of the sample files on your computer.

SQL

```
USE WideWorldImporters;
GO
BULK INSERT myTestSkipCol
    FROM 'C:\myTestSkipCol2.dat'
    WITH (FORMATFILE = 'C:\myTestSkipCol2.fmt');
GO
SELECT * FROM myTestSkipCol;
GO
```

Option #2 - Use an XML format file

Step #1 - Create a default XML format file

Create a default XML format file for the `myTestSkipCol` sample table by running the following `bcp` command at the command prompt:

Windows Command Prompt

```
bcp WideWorldImporters..myTestSkipCol format nul -f
myTestSkipCol_Default.xml -c -x -T
```

Important

You might have to specify the name of the server instance to which you are connecting with the `-s` argument. Also, you might have to specify the user name and password with the `-u` and `-P` arguments. For more information, see [bcp Utility](#).

The previous command creates an XML format file, `myTestSkipCol_Default.xml`. This format file is called a *default format file* because it is the form generated by **bcp**. A default format file describes a one-to-one correspondence between data-file fields and table columns.

XML

```
<?xml version="1.0"?>
<BCPFORMAT xmlns="http://schemas.microsoft.com/sqlserver/2004/bulkload
/format" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<RECORD>
  <FIELD ID="1" xsi:type="CharTerm" TERMINATOR="\t" MAX_LENGTH="7"/>
  <FIELD ID="2" xsi:type="CharTerm" TERMINATOR="\t" MAX_LENGTH="100"
COLLATION="SQL_Latin1_General_CP1_CI_AS"/>
  <FIELD ID="3" xsi:type="CharTerm" TERMINATOR="\r\n" MAX_LENGTH="100"
COLLATION="SQL_Latin1_General_CP1_CI_AS"/>
</RECORD>
<ROW>
  <COLUMN SOURCE="1" NAME="Col1" xsi:type="SQLSMALLINT"/>
  <COLUMN SOURCE="2" NAME="Col2" xsi:type="SQLNVARCHAR"/>
  <COLUMN SOURCE="3" NAME="Col3" xsi:type="SQLNVARCHAR"/>
</ROW>
</BCPFORMAT>
```

📌 Note

For information about the structure of XML format files, see [XML Format Files \(SQL Server\)](#).

Step #2 - Modify an XML format file

Here is the modified XML format file, `myTestSkipCol2.xml`, which skips `col2`. The `FIELD` and `ROW` entries for `col2` have been removed and the entries have been renumbered. The delimiter after the first field has also been changed from `\t` to `, .`

XML

```
<?xml version="1.0"?>
<BCPFORMAT xmlns="http://schemas.microsoft.com/sqlserver/2004/bulkload
/format" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<RECORD>
  <FIELD ID="1" xsi:type="CharTerm" TERMINATOR="," MAX_LENGTH="7"/>
  <FIELD ID="2" xsi:type="CharTerm" TERMINATOR="\r\n" MAX_LENGTH="100"
COLLATION="SQL_Latin1_General_CP1_CI_AS"/>
</RECORD>
```

```
<ROW>  
  <COLUMN SOURCE="1" NAME="Col1" xsi:type="SQLSMALLINT"/>  
  <COLUMN SOURCE="2" NAME="Col3" xsi:type="SQLNVARCHAR"/>  
</ROW>  
</BCPFORMAT>
```

Examples with an XML format file

The following examples are based on the `myTestSkipCol` sample table and the `myTestSkipCol2.dat` sample data file that are described earlier in this article.

To import the data from `myTestSkipCol2.dat` into the `myTestSkipCol` table, the examples use the modified XML format file, `myTestSkipCol2.xml`.

Use BULK INSERT with a view

With an XML format file, you cannot skip a column when you are importing directly into a table by using a `bcp` command or a `BULK INSERT` statement. However, you can import into all but the last column of a table. If you have to skip any column other than the last column, you must create a view of the target table that contains only the columns contained in the data file. Then, you can bulk import data from that file into the view.

The following example creates the `v_myTestSkipCol` view on the `myTestSkipCol` table. This view skips the second table column, `col2`. The example then uses `BULK INSERT` to import the `myTestSkipCol2.dat` data file into this view.

In SSMS, run the following code. Update the file system paths for the location of the sample files on your computer.

```
SQL
```

```
USE WideWorldImporters;
GO

CREATE VIEW v_myTestSkipCol AS
    SELECT Col1,Col3
    FROM myTestSkipCol;
GO

BULK INSERT v_myTestSkipCol
FROM 'C:\myTestSkipCol2.dat'
WITH (FORMATFILE='C:\myTestSkipCol2.xml');
GO
```

Use OPENROWSET(BULK...)

To use an XML format file to skip a table column by using `OPENROWSET(BULK...)`, you have to provide an explicit list of columns in the select list and also in the target table, as follows:

SQL

```
INSERT ...<column_list> SELECT <column_list> FROM OPENROWSET(BULK...)
```

The following example uses the `OPENROWSET` bulk rowset provider and the `myTestSkipCol2.xml` format file. The example bulk imports the `myTestSkipCol2.dat` data file into the `myTestSkipCol` table. The statement contains an explicit list of columns in the select list and also in the target table, as required.

In SSMS, run the following code. Update the file system paths for the location of the sample files on your computer.

SQL

```
USE WideWorldImporters;
GO
INSERT INTO myTestSkipCol
    (Col1,Col3)
    SELECT Col1,Col3
    FROM OPENROWSET(BULK 'C:\myTestSkipCol2.Dat',
        FORMATFILE='C:\myTestSkipCol2.Xml'
    ) as t1 ;
GO
```


Next steps

- [bcp Utility](#)
- [BULK INSERT \(Transact-SQL\)](#)
- [OPENROWSET \(Transact-SQL\)](#)
- [Use a Format File to Skip a Data Field \(SQL Server\)](#)
- [Use a Format File to Map Table Columns to Data-File Fields \(SQL Server\)](#)
- [Use a Format File to Bulk Import Data \(SQL Server\)](#)