Microsoft

Data Migration Jumpstart

Bug Free Polybase Load

21-Aug-2018

Bug Free Polybase Load to SQL DW Approach

This guide covers an approach for a bug free Polybase Load using Azure Data Factory (ADF). The following caveats apply:

- You need to guarantee the files being extracted are UTF-8/UTF-16 compatible (you can follow this white paper to achieve this)
- You need to guarantee NULL values are handled correctly – use empty string

Assumptions: Familiarity with Polybase and File Formats in SQL DW

Step 0 – Defining the File Format

A suggestion of a File Format that worked for several DMJ engagements is:

```
CREATE EXTERNAL FILE FORMAT FileFormat3

WITH

(            FORMAT_TYPE = DELIMITEDTEXT

,            FORMAT_OPTIONS    (                FIELD_TERMINATOR = ' ¬'

                               ,                STRING_DELIMITER = ' '

                               ,                USE_TYPE_DEFAULT = TRUE

                )

);
```

Step 1 – Preparing the **source data**

One of the main reasons why a table load will fail is due to carriage return and line feed in the source data. The approach taken in this guide is to remove carriage return and line feed in the source data by generating a query that will replace those characters to empty string.

Source = Oracle

```
SELECT  CASE WHEN COLUMN_ID = 1 THEN 'SELECT ' ||

          CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME || ','

         ELSE 'REPLACE(REPLACE(' || COLUMN_NAME || ', CHR(10), ''''), CHR(13), '''') AS ' || COLUMN_NAME || ','

         END

     WHEN (COLUMN_ID = (SELECT COUNT(1) FROM dba_tab_columns B where A.table_name = B.TABLE_NAME AND
A.OWNER = B.OWNER)) THEN

         CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME || ' FROM ' || OWNER || '.' || TABLE_NAME || ';'

          ELSE 'REPLACE(REPLACE(' || COLUMN_NAME || ', CHR(10), ''''), CHR(13), '''') AS ' || COLUMN_NAME  || ' FROM ' ||
OWNER || '.' || TABLE_NAME || ';'

          END

      ELSE    CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME || ','

            ELSE 'REPLACE(REPLACE(' || COLUMN_NAME || ', CHR(10), ''''), CHR(13), '''') AS ' || COLUMN_NAME || ','

            END

      END AS MEGA_QUERY
```

```
FROM      DBA_TAB_COLUMNS A
WHERE     OWNER = 'DS_SAP'
--AND     TABLE_NAME = 'DS_EMPLOYEE'
ORDER BY COLUMN_ID ASC;
```

Source = <IMPLEMENTS INFORMATION_SCHEMA>: MySQL, PostgreSQL, SQL Server, Netezza, etc

```
SELECT  CASE WHEN ORDINAL_POSITION = 1 THEN 'SELECT ' +
            CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME + ','
           ELSE 'REPLACE(REPLACE(' + COLUMN_NAME + ', CHAR(10), ''''), CHAR(13), '''') AS ' + COLUMN_NAME + ','
           END
                        WHEN (ORDINAL_POSITION = (SELECT COUNT(1) FROM INFORMATION_SCHEMA.COLUMNS B WHERE A.TABLE_SCHEMA = B.TABLE_SCHEMA AND A.TABLE_NAME = B.TABLE_NAME)) THEN
            CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME + ' FROM ' +  TABLE_SCHEMA +  '.' +  TABLE_NAME +  ';'
           ELSE 'REPLACE(REPLACE(' + COLUMN_NAME + ', CHAR(10), ''''), CHAR(13), '''') AS ' + COLUMN_NAME +  ' FROM ' + TABLE_SCHEMA +  '.' +  TABLE_NAME +  ';'
           END
                        ELSE
            CASE WHEN DATA_TYPE = 'DATE' THEN COLUMN_NAME + ','
           ELSE 'REPLACE(REPLACE(' + COLUMN_NAME + ', CHAR(10), ''''), CHAR(13), '''') AS ' + COLUMN_NAME + ','
           END
                        END
from INFORMATION_SCHEMA.COLUMNS A
WHERE TABLE_SCHEMA = 'DM_HR'
--AND TABLE_NAME = 'FCT_EMPLOYEE_ABSENCE'
ORDER BY ORDINAL_POSITION ASC
```

Sample result:

```
SELECT REPLACE(REPLACE(EMPLOYEE_SKEY, CHR(10), ''), CHR(13), '') AS EMPLOYEE_SKEY,
VALID_FROM,
VALID_TO,
REPLACE(REPLACE(COST_CENTER, CHR(10), ''), CHR(13), '') AS COST_CENTER,
REPLACE(REPLACE(ORG_UNIT_NAME, CHR(10), ''), CHR(13), '') AS ORG_UNIT_NAME,
REPLACE(REPLACE(PARENT_ORG_UNIT, CHR(10), ''), CHR(13), '') AS PARENT_ORG_UNIT,
REPLACE(REPLACE(STG_LOAD_DELTA_ID, CHR(10), ''), CHR(13), '') AS STG_LOAD_DELTA_ID,
STG_LOAD_DATE FROM DS_SAP.DS_EMPLOYEE;
```

Step 2 – Preparing the **target External table in SQL DW**

Another reason a table load will fail is due to datatype mismatch and irregular datatype conversions from source to target. One way of bypassing this situation is by creating the External Table with generic datatypes. In a nutshell, the technique consists in converting all datatypes to varchar. That means, we will be reading the external tables as pure texts and won't be worried with datatype conversion for now.

As an example, this is what the target database table looks like in SQL DW

```
CREATE TABLE [dm_hr].[FCT_EMPLOYEE_ABSENCE]
(
	[FCT_EMPLOYEE_ABSENCE_DWK] [bigint] IDENTITY(1,1) NOT NULL,
	[DIM_EMPLOYEE_DWK] [bigint] NOT NULL,
	[CALENDAR_DWK] [bigint] NOT NULL,
	[DIM_TIME_TYPE_DWK] [bigint] NOT NULL,
	[DIM_UNIT_OF_MEASUREMENT_DWK] [bigint] NOT NULL,
	[DIM_ORGANISATION_DWK] [bigint] NOT NULL,
	[ABSENCE_ACTUAL_TIME] [numeric](38, 1) NULL,
	[DM_LOAD_DELTA_ID] [numeric](38,1) NULL,
	[DM_NAME] [varchar(50)] NULL,
	[DM_LOAD_DATE] [datetime2](0) NULL
)
WITH
(
	DISTRIBUTION = ROUND_ROBIN,
	CLUSTERED COLUMNSTORE INDEX
)
GO
```

This is what the external table will look like in SQL DW:

As I said previously, the use of generic datatypes saves time with datatype conversion problems. Meaning, there will be always only have 3 kinds of datatypes in the external tables: Integer/Bigint, Datetime2(7) and varchar(XX). If you want, for simplicity, you can convert everything to Varchar.

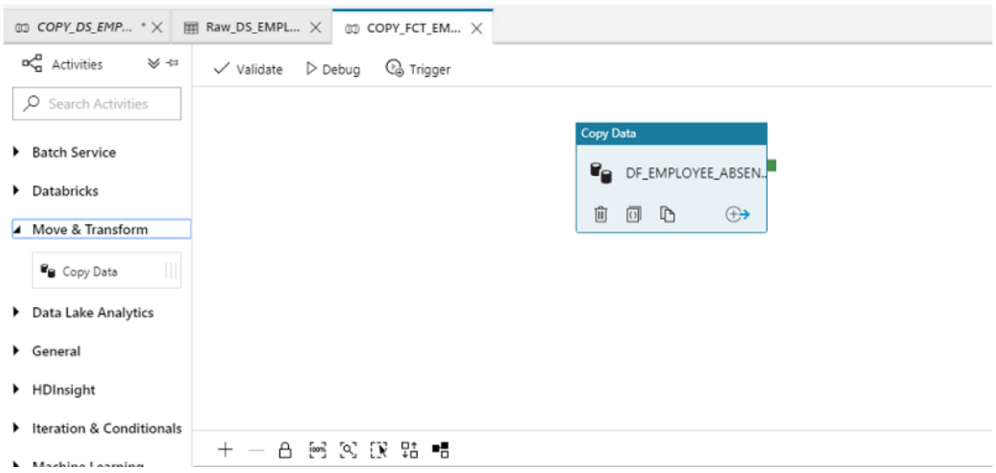
Sample external table in SQL DW:

```
CREATE EXTERNAL TABLE [dm_hr].[ext_FCT_EMPLOYEE_ABSENCE]
(
	[FCT_EMPLOYEE_ABSENCE_DWK] [bigint] NULL,
	[DIM_EMPLOYEE_DWK] [bigint] NULL,
	[CALENDAR_DWK] [bigint] NULL,
	[DIM_TIME_TYPE_DWK] [bigint] NULL,
	[DIM_UNIT_OF_MEASUREMENT_DWK] [bigint] NULL,
	[DIM_ORGANISATION_DWK] [bigint] NULL,
	[ABSENCE_ACTUAL_TIME] [varchar](39) NULL,
	[DM_LOAD_DELTA_ID] [varchar](39) NULL,
	[DM_NAME] [varchar(50)] NULL,
	[DM_LOAD_DATE] [datetime2](7) NULL
)
WITH (
DATA_SOURCE = [ds_adl],
LOCATION = N'/Raw/DM_HR/FCT_EMPLOYEE_ABSENCE',
FILE_FORMAT = [FileFormat3],
REJECT_TYPE = VALUE,
REJECT_VALUE = 1000
)
GO
```

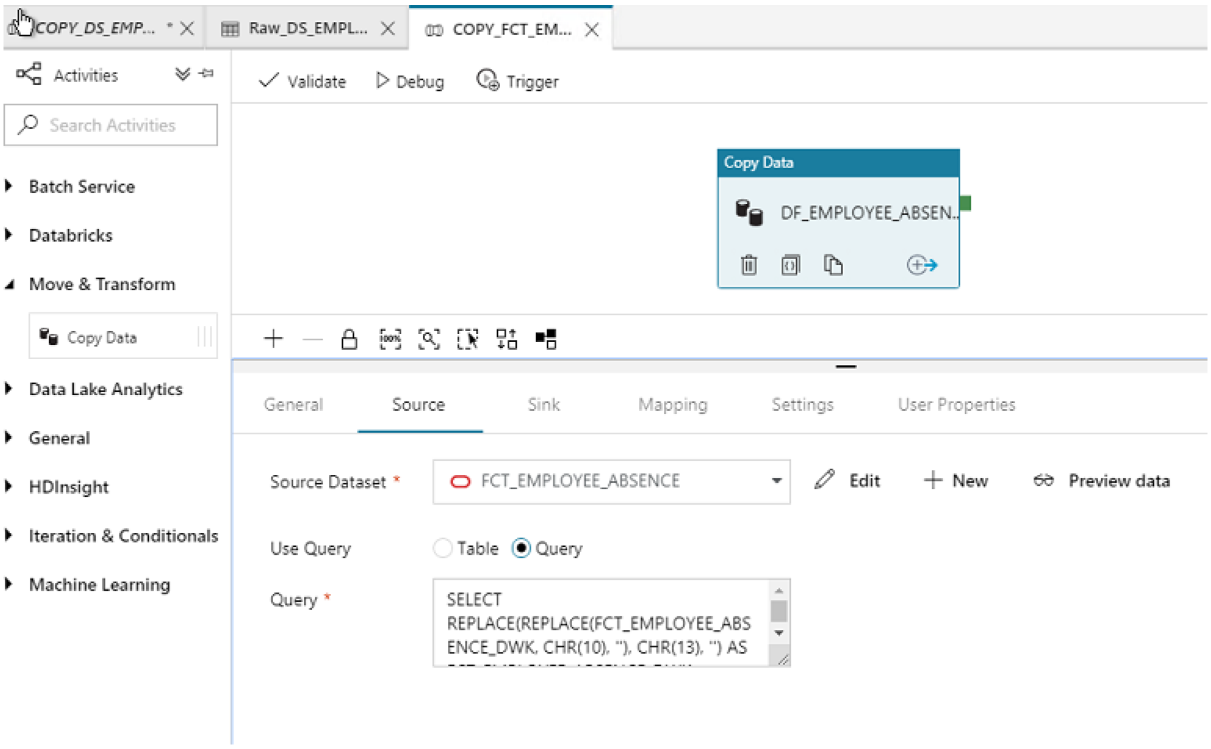
Step 3 – **ADF configuration**

Make sure the following setup is being used in ADF:

Sample Copy data Pipeline:



Go to "Source" and make sure you check "Use Query" -> "Query". Here you paste the query generated on Step 1 for that table in the "Query" box.



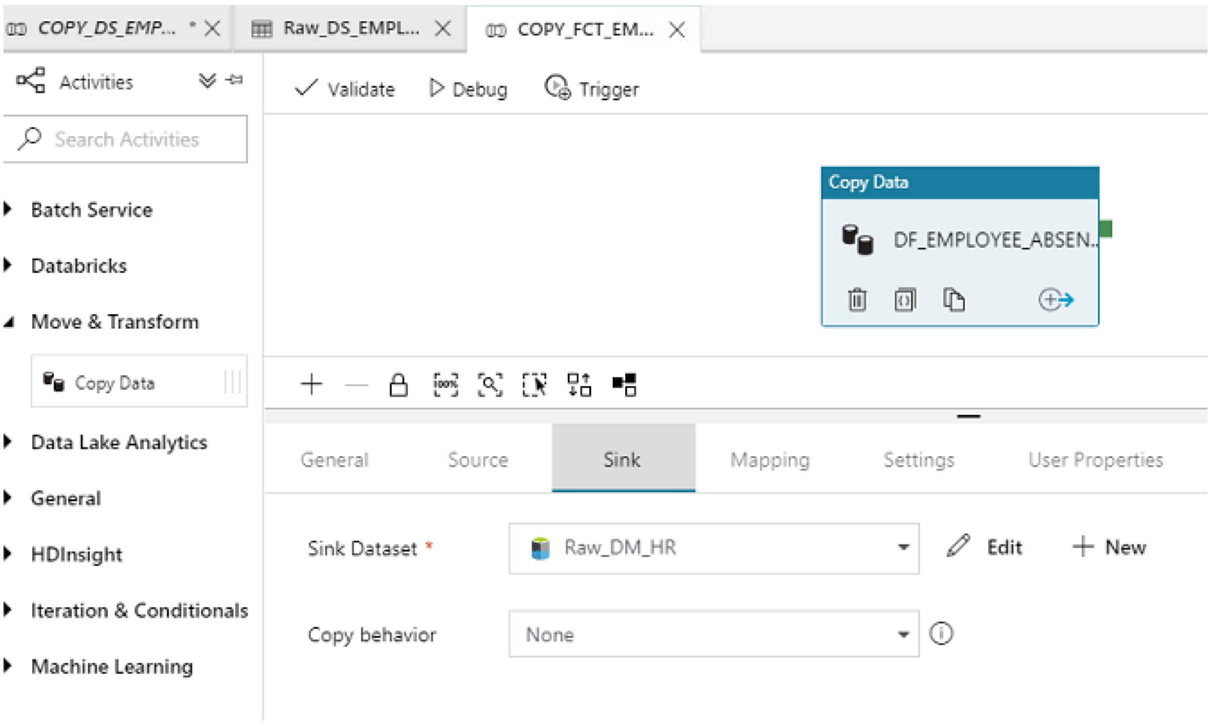
Source Dataset * FCT_EMPLOYEE_ABSENCE

Use Query ○ Table ● Query

Query *

```
SELECT
REPLACE(REPLACE(FCT_EMPLOYEE_ABS
ENCE_DWK, CHR(10), ''), CHR(13), '') AS
```

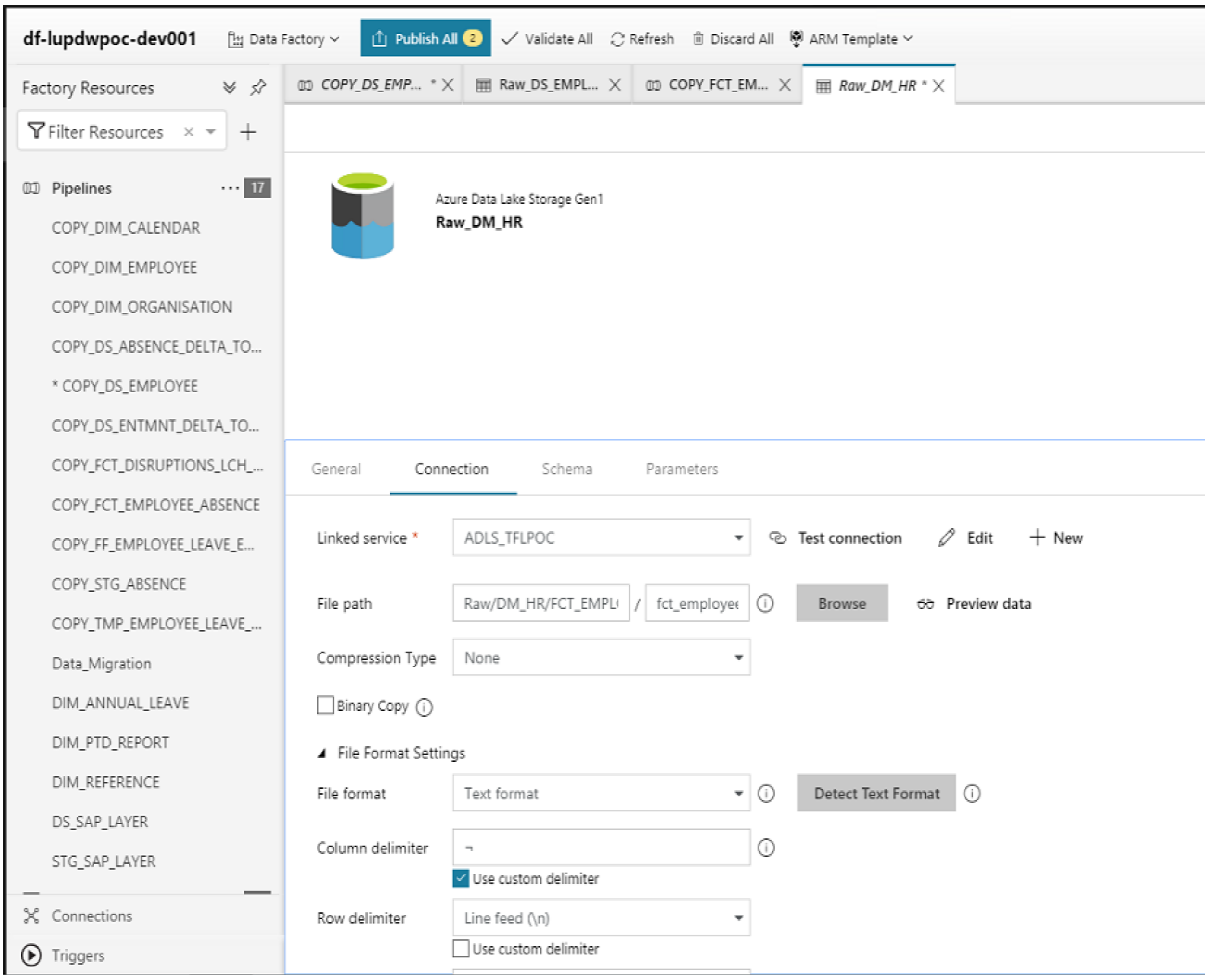
After that you go click on "Sink" -> "Edit"



Sink Dataset * : Raw_DM_HR

Copy behavior: None

Navigate to "Connection" and make sure you configure it as following:



df-lupdwpoc-dev001

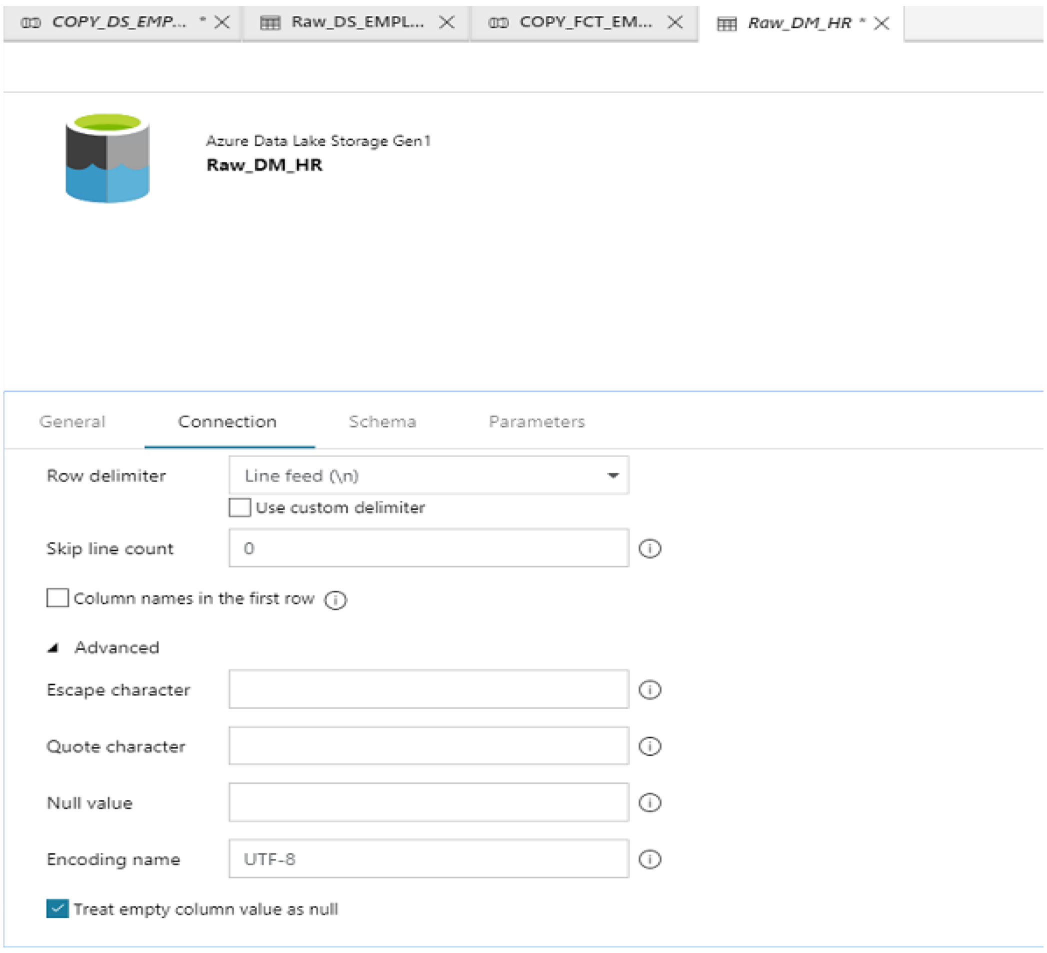
Factory Resources

Pipelines 17

- COPY_DIM_CALENDAR
- COPY_DIM_EMPLOYEE
- COPY_DIM_ORGANISATION
- COPY_DS_ABSENCE_DELTA_TO...
- * COPY_DS_EMPLOYEE
- COPY_DS_ENTMNT_DELTA_TO...
- COPY_FCT_DISRUPTIONS_LCH_...
- COPY_FCT_EMPLOYEE_ABSENCE
- COPY_FF_EMPLOYEE_LEAVE_E...
- COPY_STG_ABSENCE
- COPY_TMP_EMPLOYEE_LEAVE_...
- Data_Migration
- DIM_ANNUAL_LEAVE
- DIM_PTD_REPORT
- DIM_REFERENCE
- DS_SAP_LAYER
- STG_SAP_LAYER

Azure Data Lake Storage Gen1
Raw_DM_HR

Linked service *	ADLS_TFLPOC
File path	Raw/DM_HR/FCT_EMPL / fct_employee
Compression Type	None
Binary Copy	☐
File format	Text format
Column delimiter	¬ ☑ Use custom delimiter
Row delimiter	Line feed (\n) ☐ Use custom delimiter


COPY_DS_EMP... * | Raw_DS_EMPL... | COPY_FCT_EM... | Raw_DM_HR *

Azure Data Lake Storage Gen1
Raw_DM_HR

General | Connection | Schema | Parameters

Row delimiter: Line feed (\n)
☐ Use custom delimiter
Skip line count: 0
☐ Column names in the first row
Advanced
Escape character:
Quote character:
Null value:
Encoding name: UTF-8
☑ Treat empty column value as null

File Format: TextFormat

Column Delimiter : ¬ (tick "use custom delimiter" to enable that)

Row Delimiter: Line Feed (\n)

Click in "Advanced" to expand the window

Null Value : Make sure you renove the default (\N) and leave it empty

Encoding name: UTF-8

And tick "Treat empty column value as null"

There might have other configurations that you want to do, such as quote character, escape character etc. What it is showed here is the essential configuration that needs to be in place for this approach to work.

Step 4 – Insert into **SQL DW target table** from the SQL DW External table

Lastly, generate the procedure that will load the data from the External SQL DW table to the Internal SQL DW table. Now, the idea is to cast/convert the datatypes to fit the target SQL DW table.

Again, the technique consists in generating the columns casting using the DBMS metadata. See how it is done as following:

```
SELECT  CASE WHEN ORDINAL_POSITION = 1 THEN 'SELECT ' +
                     CASE
                     WHEN DATA_TYPE IN ('bigint', 'int', 'smallint', 'float') AND
IS_NULLABLE='YES' then 'cast('+column_name+' as ' + data_type +'),'
                     WHEN DATA_TYPE='date' AND IS_NULLABLE='YES' then
'cast('+column_name+' as date),'
                     WHEN DATA_TYPE='datetime2' AND IS_NULLABLE='YES' then
'convert(datetime2(0), CAST('+column_name+' AS DATETIME2(7)), 103),'
                     WHEN DATA_TYPE='varchar'  AND IS_NULLABLE='YES' then
'cast('+column_name+' as varchar(' + convert(varchar, character_maximum_length) +
')),'
                     WHEN DATA_TYPE='numeric' AND IS_NULLABLE='YES' then
'CONVERT(numeric('+ convert(varchar, numeric_precision) + ',' + convert(varchar,
numeric_scale) + '), CASE WHEN ISNUMERIC(' + column_name+ ')=1 THEN cast(' +
column_name + ' as numeric('+ convert(varchar, numeric_precision) + ',' +
convert(varchar, numeric_scale) +')) ELSE null END),'
                     ELSE COLUMN_NAME + ','
                     end
              WHEN (ORDINAL_POSITION = (SELECT COUNT(1) FROM
INFORMATION_SCHEMA.COLUMNS B WHERE A.TABLE_SCHEMA = B.TABLE_SCHEMA AND A.TABLE_NAME =
B.TABLE_NAME)) THEN
                     CASE
                     WHEN DATA_TYPE IN ('bigint', 'int', 'smallint', 'float') AND
IS_NULLABLE='YES' then 'cast('+column_name+' as ' + data_type +') FROM ' +
TABLE_SCHEMA + '.' + TABLE_NAME +  ';'
                     WHEN DATA_TYPE='date' AND IS_NULLABLE='YES' then
'cast('+column_name+' as date) FROM ' + TABLE_SCHEMA + '.' + TABLE_NAME +  ';'
                     WHEN DATA_TYPE='datetime2' AND IS_NULLABLE='YES' then
'convert(datetime2(0), CAST('+column_name+' AS DATETIME2(7)), 103) FROM ' +
TABLE_SCHEMA + '.' + TABLE_NAME +  ';'
                     WHEN DATA_TYPE='varchar'  AND IS_NULLABLE='YES' then
'cast('+column_name+' as varchar(' + convert(varchar, character_maximum_length) + '))
FROM ' + TABLE_SCHEMA + '.' + TABLE_NAME +  ';'
                     WHEN DATA_TYPE='numeric' AND IS_NULLABLE='YES' then
'CONVERT(numeric('+ convert(varchar, numeric_precision) + ',' + convert(varchar,
numeric_scale) + '), CASE WHEN ISNUMERIC(' + column_name+ ')=1 THEN cast(' +
column_name + ' as numeric('+ convert(varchar, numeric_precision) + ',' +
convert(varchar, numeric_scale) +')) ELSE null END) FROM ' + TABLE_SCHEMA + '.' +
TABLE_NAME +  ';'
                     ELSE COLUMN_NAME + ' FROM ' + TABLE_SCHEMA + '.' + TABLE_NAME +
';'
                     END
              ELSE
                     CASE
                     WHEN DATA_TYPE IN ('bigint', 'int', 'smallint', 'float') AND
IS_NULLABLE='YES' then 'cast('+column_name+' as ' + data_type +'),'
                     WHEN DATA_TYPE='date' AND IS_NULLABLE='YES' then
'cast('+column_name+' as date),'
                     WHEN DATA_TYPE='datetime2' AND IS_NULLABLE='YES' then
'convert(datetime2(0), CAST('+column_name+' AS DATETIME2(7)), 103),'
                     WHEN DATA_TYPE='varchar'  AND IS_NULLABLE='YES' then
'cast('+column_name+' as varchar(' + convert(varchar, character_maximum_length) +
')),'
                     WHEN DATA_TYPE='numeric' AND IS_NULLABLE='YES' then
'CONVERT(numeric('+ convert(varchar, numeric_precision) + ',' + convert(varchar,
numeric_scale) + '), CASE WHEN ISNUMERIC(' + column_name+ ')=1 THEN cast(' +
column_name + ' as numeric('+ convert(varchar, numeric_precision) + ',' +
convert(varchar, numeric_scale) +')) ELSE null END),'
                     ELSE COLUMN_NAME + ','
                     END
              END
```

```
from INFORMATION_SCHEMA.COLUMNS A WHERE TABLE_NAME = 'FCT_EMPLOYEE_ABSENCE'  and
TABLE_SCHEMA = 'DM_HR' ORDER BY ORDINAL_POSITION ASC;
```

Sample output for a single table:

SELECT FCT_EMPLOYEE_ABSENCE_DWK,

DIM_EMPLOYEE_DWK,

CALENDAR_DWK,

DIM_TIME_TYPE_DWK,

DIM_UNIT_OF_MEASUREMENT_DWK,

DIM_ORGANISATION_DWK,

CONVERT(numeric(38,0), CASE WHEN ISNUMERIC(ABSENCE_ACTUAL_TIME)=1 THEN cast(ABSENCE_ACTUAL_TIME as numeric(38,0)) ELSE null END),

CONVERT(numeric(38,0), CASE WHEN ISNUMERIC(DM_LOAD_DELTA_ID)=1 THEN cast(DM_LOAD_DELTA_ID as numeric(38,0)) ELSE null END),

convert(datetime2(0), CAST(DM_LOAD_DATE AS DATETIME2(7)), 103) FROM dm_hr.FCT_EMPLOYEE_ABSENCE;

Sample stored procedure:

```
CREATE PROC [dm_hr].[spLoad_FCT_EMPLOYEE_ABSENCE] AS
SET NOCOUNT ON;
BEGIN
TRUNCATE TABLE dm_hr.FCT_EMPLOYEE_ABSENCE;

SET IDENTITY_INSERT dm_hr.FCT_EMPLOYEE_ABSENCE ON

INSERT INTO dm_hr.FCT_EMPLOYEE_ABSENCE(
FCT_EMPLOYEE_ABSENCE_DWK,
DIM_EMPLOYEE_DWK,
CALENDAR_DWK,
DIM_TIME_TYPE_DWK,
DIM_UNIT_OF_MEASUREMENT_DWK,
DIM_ORGANISATION_DWK,
ABSENCE_ACTUAL_TIME,
DM_LOAD_DELTA_ID,
DM_LOAD_DATE)
SELECT
FCT_EMPLOYEE_ABSENCE_DWK,
DIM_EMPLOYEE_DWK,
CALENDAR_DWK,
DIM_TIME_TYPE_DWK,
DIM_UNIT_OF_MEASUREMENT_DWK,
DIM_ORGANISATION_DWK,
CONVERT(numeric(38,0), CASE WHEN ISNUMERIC(ABSENCE_ACTUAL_TIME)=1 THEN
cast(ABSENCE_ACTUAL_TIME as numeric(38,0)) ELSE null END),
CONVERT(numeric(38,0), CASE WHEN ISNUMERIC(DM_LOAD_DELTA_ID)=1 THEN
cast(DM_LOAD_DELTA_ID as numeric(38,0)) ELSE null END),
convert(datetime2(0), CAST(DM_LOAD_DATE AS DATETIME2(7)), 103)
FROM dm_hr.ext_FCT_EMPLOYEE_ABSENCE

SET IDENTITY_INSERT dm_hr.FCT_EMPLOYEE_ABSENCE OFF
```

```
END
GO
```

Data Load:

```
Exec dm_hr.spLoad_FCT_EMPLOYEE_ABSENCE
```

Feedback and suggestions

If you have feedback or suggestions for improving this data migration asset, please contact the Data Migration Jumpstart Team (askdmjfordmtools@microsoft.com). Thanks for your support!

Note: For additional information about migrating various source databases to Azure, see the Azure Database Migration Guide.