**SSIS Data Conversion Transformation – Trainer Notes (Excel → SQL, Unicode → Non‑Unicode)**

# 1) What is the Data Conversion Transformation?

Data Conversion is a Data Flow transformation that changes the data type, length, scale, code page, or precision of columns as rows pass through the pipeline. It is commonly used to resolve type mismatches between sources (e.g., Excel’s Unicode text) and destinations (e.g., SQL Server VARCHAR).

# 2) When to Use It (and When Not To)

* Use it when moving from Unicode (DT\_WSTR) to non‑Unicode (DT\_STR), or between numeric types (e.g., DT\_R8 → DT\_NUMERIC).
* Use it when Excel/CSV/flat file types don’t match your SQL table column types.
* Prefer doing conversions in the source SQL (CAST/CONVERT) when reading from a database for pushdown performance.
* For string cleanup (TRIM/REPLACE), add a Derived Column after (or instead of) Data Conversion.

# 3) Unicode vs. Non‑Unicode Basics (Excel & SQL Server)

• Excel Source emits text as DT\_WSTR (Unicode).  
• SQL Server NVARCHAR corresponds to DT\_WSTR; VARCHAR corresponds to DT\_STR (with a code page, usually 1252 for Latin).  
• The classic error: “Cannot convert between Unicode and non‑Unicode string data types” appears when mapping DT\_WSTR to a VARCHAR destination without converting.

## Quick Mapping Cheat‑Sheet

|  |  |  |  |
| --- | --- | --- | --- |
| **Source (Excel/Flat)** | **SSIS Type** | **Convert To (Data Conversion)** | **SQL Server Type** |
| Text (Unicode) | DT\_WSTR (length) | DT\_STR (length, CodePage=1252) | VARCHAR(length) |
| Text (Unicode) | DT\_WSTR (length) | — (no change) | NVARCHAR(length) |
| Number (double) | DT\_R8 | DT\_NUMERIC (precision, scale) or DT\_I4/DT\_I8 | DECIMAL(p,s) / INT / BIGINT |
| Date | DT\_DATE / DT\_DBTIMESTAMP | — (or DT\_DBTIMESTAMP) | DATE/DATETIME/DATETIME2 |

# 4) Lab Setup (Destination Table and Example Excel)

## SQL Destination Table (VARCHAR columns)

```sql  
CREATE TABLE dbo.TestData (

ID INT NULL,

first\_name VARCHAR(50) NULL,

last\_name VARCHAR(50) NULL,

gender VARCHAR(20) NULL,

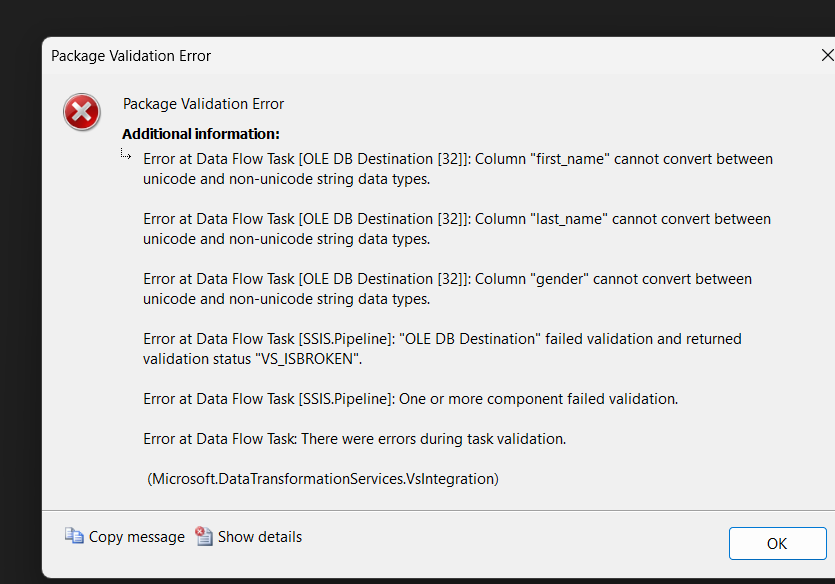
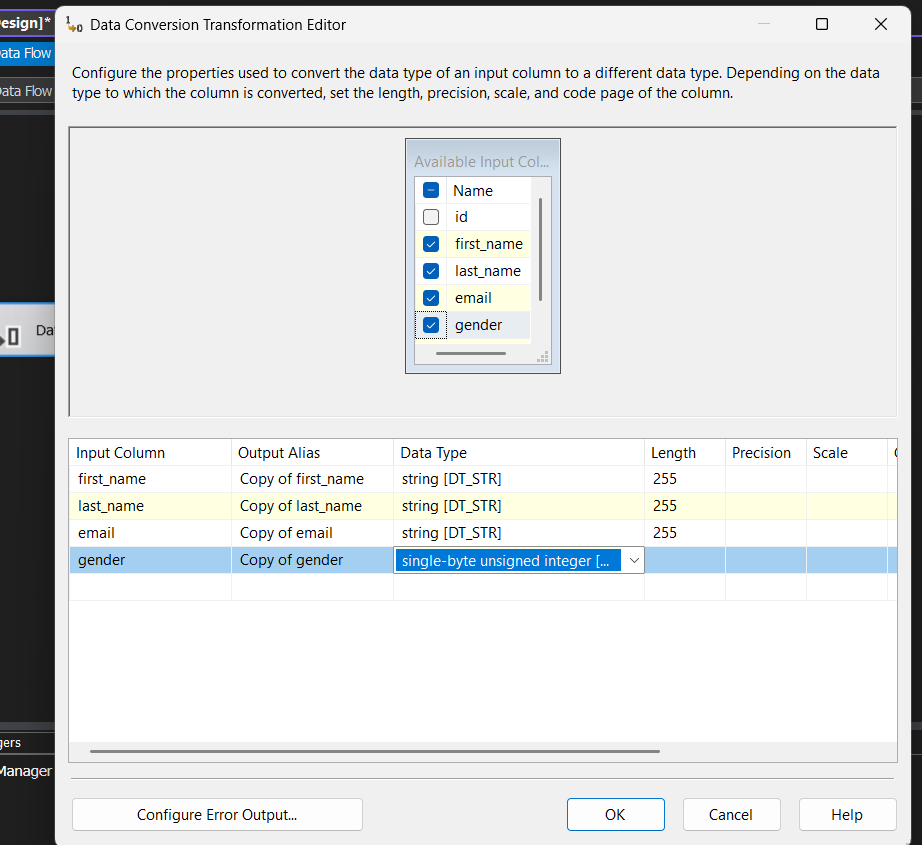
email VARCHAR(100) NULL

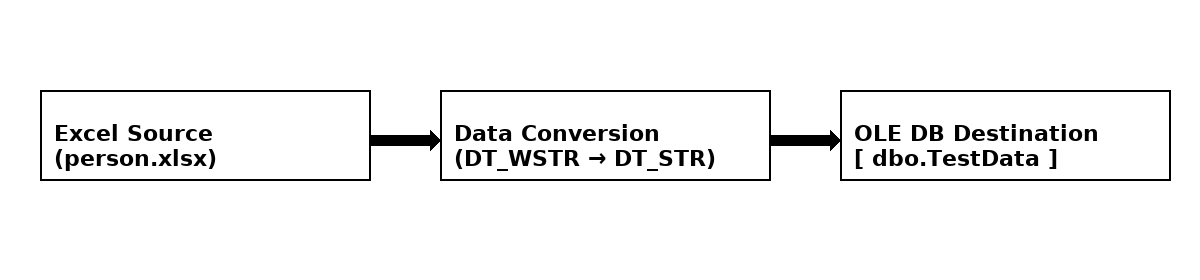
);

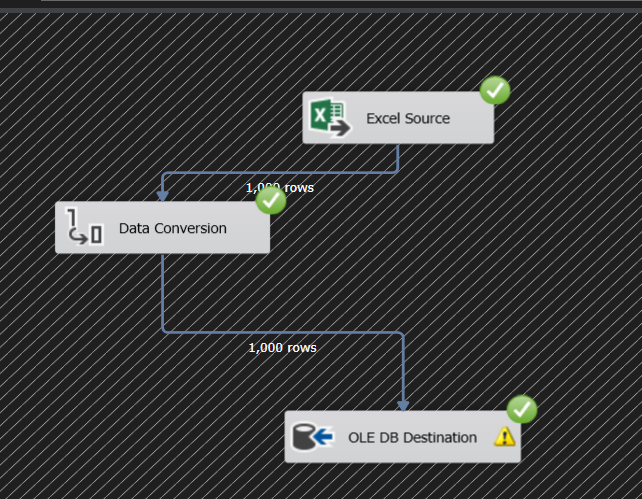
```

Excel: worksheet has columns ID, first\_name, last\_name, gender, company\_name (all text except ID).

# 5) Step‑by‑Step in Visual Studio 2022 (Excel → SQL, Unicode → VARCHAR)

1. Create an Integration Services Project and add a Data Flow Task.
2. Inside the Data Flow, add Excel Source. Create an Excel Connection to your person.xlsx, select the worksheet, and tick First row has column names.
3. Add an OLE DB Destination pointing to your database and choose dbo.TestData. Open Mappings. You will likely see the red error icon complaining about Unicode → non‑Unicode for string columns.
4. 
5. Remove the direct link to the destination. Add a Data Conversion transform between Excel Source and OLE DB Destination.
6. Open Data Conversion and configure each string column:  
    • Input Column: first\_name → Data Type: string [DT\_STR], Length: 50, Code Page: 1252; Output Alias: first\_name\_varchar  
    • Input Column: last\_name → Data Type: DT\_STR, Length: 50, Code Page: 1252; Output Alias: last\_name\_varchar  
    • Input Column: gender → Data Type: DT\_STR, Length: 20, Code Page: 1252; Output Alias: gender\_varchar  
    • Input Column: company\_name → Data Type: DT\_STR, Length: 100, Code Page: 1252; Output Alias: company\_varchar
7. 
8. Reconnect Data Conversion → OLE DB Destination. In Mappings, map the destination columns to the converted aliases (e.g., first\_name ← first\_name\_varchar).
9. Run the package. Rows should now load without the Unicode error.





# 6) Excel Driver (ACE) and 64‑bit vs 32‑bit Runtime

* If you get “Class not registered” or acquire-connection errors with Excel Source during debugging, set Project → Properties → Configuration Properties → Debugging → Run64BitRuntime = False and run again.
* When scheduling in SQL Agent, choose Use 32-bit runtime on the job step if the 64-bit ACE OLE DB provider is not installed.
* Alternatively, install the appropriate Microsoft Access Database Engine (ACE) provider matching your runtime.

# 7) Troubleshooting Guide

* Cannot convert between Unicode and non‑Unicode string data types → Add Data Conversion for DT\_WSTR → DT\_STR (or change destination to NVARCHAR).
* Potential loss of data → Increase the Length in Data Conversion (and/or widen the destination column).
* Truncation errors → In Data Conversion, set the output length to match or exceed the destination; avoid silent truncation by reviewing warnings.
* Wrong characters after load → Set Code Page = 1252 (or appropriate) when converting to DT\_STR.
* Mixed data in Excel columns (nulls) → Excel type guessing can treat a mostly-numeric column as numeric and null out text. Force the column as text in Excel, or read as Unicode text and convert later.
* Headers inserted as data → Ensure Excel Source option First row has column names is checked.
* Slow conversions → When source is SQL Server, prefer doing CAST/CONVERT in the source query to push work to the database engine.

# 8) Advanced Patterns & Variations

* Convert numbers arriving as text: Use Data Conversion to convert DT\_WSTR → DT\_I4/DT\_R8 as needed, then map to INT/DECIMAL columns.
* Chain Derived Column after Data Conversion to TRIM/UPPER/REPLACE, or to handle NULLs with (DT\_STR,50,1252)(ISNULL(col)?"":col) patterns.
* If destination is NVARCHAR, you can skip conversion (DT\_WSTR maps directly) and only use Derived Column for cleanup.

# 9) Verification Queries

```sql  
-- Loaded row count  
SELECT COUNT(\*) AS rows\_loaded FROM dbo.TestData;  
  
-- Spot-check data lengths to detect truncation risks  
SELECT MAX(LEN(first\_name)) AS max\_first, MAX(LEN(last\_name)) AS max\_last,  
 MAX(LEN(company\_name)) AS max\_company  
FROM dbo.TestData;  
```

# 10) Quick Reference (UI Settings)

* Excel Source: choose worksheet, check First row has column names.
* Data Conversion: set Data Type = DT\_STR, Length to match VARCHAR size, Code Page = 1252; give clear Output Alias names (e.g., \_varchar).
* OLE DB Destination: Mappings → map destination VARCHAR columns to the converted aliases.
* If you see red ‘X’ on the destination: it’s usually a Unicode → non‑Unicode mismatch or length mismatch.