**[SSIS Lookup Transformation Case Sensitivity](http://geekswithblogs.net/dtotzke/archive/2007/02/06/105585.aspx" \o "Title of this entry.)**

Hey there,

In one of my data loads I have NULLs creeping into a foreign key column.  I am looking up a domain name and getting an integer Id number back.  The column is populated during the load by a Lookup Transformation in SSIS.  A previous step in the process populates the lookup table with any values that don’t exist so it should be impossible for the subsequent lookups to fail.

These null values had only recently appeared and coincided with some data back-filling we were doing.

**First issue**:   
I had the Lookup set to ignore failed lookups and simply return null.  This prevented me from seeing the error.  Because the lookup table gets pre-populated, a failed lookup should have been impossible.  I changed this to redirect the row upon failed lookup and put the failed rows into a separate table for examination.  Our database is set to use case insensitive collation.  Running a T-SQL query joining the failed rows to the lookup table in management studio confirms that all of the records have a match.  Arrrgh.

**Second issue**: Further examination of the data showed that the only difference in the joined columns was case.  i.e. [www.SomeDomain.com](http://www.somedomain.com/) and [www.somedomain.com](http://www.somedomain.com/) .  There is no setting on the Lookup Transformation to ignore case.

**Root cause**:  
Many of the failing rows were happening when looking up values that had been in the database for months.  They had worked in the past.  Somebody was touching the data.

The domain name value being used to do the lookup is used at the source for display to the end user.  Somebody was making them pretty by adding a little Pascal casing to them.  (another example reinforcing the importance of the separation of data from its presentation and formatting.  This explains why what once worked, now doesn’t.

The SSIS Lookup Transformation uses full caching by default.  It pre-loads all of the lookup table data into memory at the start of the process.  In full cache mode, the lookup relies on the LocaleId for string comparison rules.  It picks up the locale of its parent container or from the package if not contained within something such as a foreach container.  This brings case and accent sensitivity into play.

What had me fooled is that when configuring the lookup you JOIN the columns in the editor the same way you would visually join columns in a query builder.  In fact, the documentation lists the Lookup under “Split and Join Transformations” and states that it “performs lookups by joining data in input columns with columns in a reference dataset”.  One would think that since the source of its data is SQL Server, it would pick up the collation setting of the database.  You’d think that; and you’d be wrong.

Of course, had I RTFM, I would have learned that the “lookups performed by the Lookup transformation are case sensitive” and been happier for the experience.

**Resolution**:

There are a few solutions to this problem depending on the size of your data loads and lookup tables.

1. Set the Lookup Transformation caching option to Partial or None.  
   Setting the cache mode to partial or none causes comparisons to be done by SQL Server.
2. Full cache with pre-formatted data.  
   Update the lookup column using the T-SQL LOWER() or UPPER() functions so that subsequent lookups done by the Lookup Transformation will succeed.
3. Full caching and T-SQL queries instead of tables as your data sources.  
   Use the LOWER() or UPPER() functions on the columns to be compared.  This is likely the best of both worlds.  Gives you the performance of  full caching as well as simplifies your data flow by removing the extra steps needed for option 2.