Temp table vs Tamp Variable

There are a few differences between Temporary Tables (#tmp) and Table Variables (@tmp), although using tempdb isn't one of them, as spelt out in the MSDN link below.

As a rule of thumb, for small to medium volumes of data and simple usage scenarios you should use table variables. (This is an overly broad guideline with of course lots of exceptions - see below and following articles.)

Some points to consider when choosing between them:

* Temporary Tables are real tables so you can do things like CREATE INDEXes, etc. If you have large amounts of data for which accessing by index will be faster then temporary tables are a good option.
* Table variables can have indexes by using PRIMARY KEY or UNIQUE constraints. (If you want a non-unique index just include the primary key column as the last column in the unique constraint. If you don't have a unique column, you can use an identity column.) [SQL 2014 has non-unique indexes too](https://stackoverflow.com/questions/886050/sql-server-creating-an-index-on-a-table-variable/17385085#17385085).
* Table variables don't participate in transactions and SELECTs are implicitly with NOLOCK. The transaction behaviour can be very helpful, for instance if you want to ROLLBACK midway through a procedure then table variables populated during that transaction will still be populated!
* Temp tables might result in stored procedures being recompiled, perhaps often. Table variables will not.
* You can create a temp table using SELECT INTO, which can be quicker to write (good for ad-hoc querying) and may allow you to deal with changing datatypes over time, since you don't need to define your temp table structure upfront.
* You can pass table variables back from functions, enabling you to encapsulate and reuse logic much easier (eg make a function to split a string into a table of values on some arbitrary delimiter).
* Using Table Variables within user-defined functions enables those functions to be used more widely (see CREATE FUNCTION documentation for details). If you're writing a function you should use table variables over temp tables unless there's a compelling need otherwise.
* Both table variables and temp tables are stored in tempdb. But table variables (since 2005) default to the collation of the current database versus temp tables which take the default collation of tempdb ([ref](https://technet.microsoft.com/en-us/library/ms188927.aspx)). This means you should be aware of collation issues if using temp tables and your db collation is different to tempdb's, causing problems if you want to compare data in the temp table with data in your database.
* Global Temp Tables (##tmp) are another type of temp table available to all sessions and users.

Some further reading:

* [Martin Smith's great answer](https://dba.stackexchange.com/a/16386) on dba.stackexchange.com
* MSDN FAQ on difference between the two: <https://support.microsoft.com/en-gb/kb/305977>
* MDSN blog article: <http://blogs.msdn.com/sqlserverstorageengine/archive/2008/03/30/sql-server-table-variable-vs-local-temporary-table.aspx>
* Article: [http://searchsqlserver.techtarget.com/tip/0,289483,sid87\_gci1267047,00.html#](http://searchsqlserver.techtarget.com/tip/0,289483,sid87_gci1267047,00.html)
* Unexpected behaviors and performance implications of temp tables and temp variables: [Paul White on SQLblog.com](https://sql.kiwi/2012/08/temporary-tables-in-stored-procedures.html)

1. Temp table: A Temp table is easy to create and back up data.

Table variable: But the table variable involves the effort when we usually create the normal tables.

1. Temp table: Temp table result can be used by multiple users.

Table variable: But the table variable can be used by the current user only.

1. Temp table: Temp table will be stored in the tempdb. It will make network traffic. When we have large data in the temp table then it has to work across the database. A Performance issue will exist.

Table variable: But a table variable will store in the physical memory for some of the data, then later when the size increases it will be moved to the tempdb.

1. Temp table: Temp table can do all the DDL operations. It allows creating the indexes, dropping, altering, etc..,

Table variable: Whereas table variable won't allow doing the DDL operations. But the table variable allows us to create the clustered index only.

1. Temp table: Temp table can be used for the current session or global. So that a multiple user session can utilize the results in the table.

Table variable: But the table variable can be used up to that program. (Stored procedure)

1. Temp table: Temp variable cannot use the transactions. When we do the DML operations with the temp table then it can be rollback or commit the transactions.

Table variable: But we cannot do it for table variable.

1. Temp table: Functions cannot use the temp variable. More over we cannot do the DML operation in the functions.

Table variable: But the function allows us to use the table variable. But using the table variable we can do that.

1. Temp table: The stored procedure will do the recompilation (can't use same execution plan) when we use the temp variable for every sub sequent calls.

Table variable: Whereas the table variable won't do like that.

