Best practice rules

Page last updated 05 February 2021

Adhere to good industry practices.

BP001 – Index type is not specified

The index type (whether it is CLUSTERED or NONCLUSTERED) is not specified explicitly in the CREATE INDEX statement.

You can only have one clustered index on a table, of course, and this choice has a lot of influence on the performance of queries, so you should take care to select wisely. The primary key is often, but not only, the correct choice.

Learn more about this rule

BP002 – ORDER BY clause with constants

Constants are being used in the ORDER BY clause.

The use of constants in the ORDER BY is deprecated for removal in the future. They make ORDER BY statements more difficult to understand.

Learn more about this rule

BP003 – SELECT in trigger

One or more SELECT statements were found in a trigger.

The trigger should never return data to a client. It is possible to place a SELECT statement in a trigger, but it serves no practical, useful purpose, and can have unexpected effects.

Learn more about this rule

BP004 – INSERT without column list

The column list for the insert statement is missing.

The INSERT statement need not have a column list, but omitting it assumes certain columns in a particular order. It likely to cause errors if the table in to which the inserts will be made is changed, particularly with table variables where insertions are not checked. Column lists also make code more intelligible.

Learn more about this rule

BP005 – Asterisk in select list

There is an asterisk instead of a column list in a SELECT statement.

*SELECT * FROM in IF EXISTS statements are fine, but SELECT * in other contexts assumes certain columns in a particular order, which may not last. Also, results should always consist of just the columns you need.*

Learn more about this rule

BP006 – TOP without ORDER BY

TOP is being used in a SELECT statement without a subsequent ORDER BY clause.

This is legal in SQL but meaningless because asking for the TOP 10 rows implies a certain order, and tables have no implicit logical order.

Learn more about this rule

BP007 – Variable length datatype without explicit length

Variable length datatype is declared without explicit length (VARCHAR, VARBINARY and NVARCHAR).

A variable length datatype that is declared without an explicit length is a shorthand for specifying a length of 1.

Learn more about this rule

BP008 – CAST/CONVERT to var types without length

The length of VARCHAR, VARBINARY and NVARCHAR datatype in a CAST or CONVERT clause wasn't explicitly specified.

When you convert a datatype to a varchar, you do not have to specify the length. If you don't do so, SQL Server will use a Varchar length sufficient to hold the string. It is better to specify the length because SQL Server has no idea what length you may subsequently need.

Learn more about this rule

BP009 – Avoid var types of length 1 or 2

You have used a Variable length datatype (NVARCHAR , VARCHAR or BINARY) of length 1 or 2.

Very small CHAR and BINARY values are more economically stored than their VAR equivalents. Specify the datatype of the fixed-length version.

Learn more about this rule

BP010 – Usage of @@IDENTITY

@@IDENTITY is being used to get the value of the last identity insertion.

If you have a trigger on the table, the value can sometimes be wrong. The @@IDENTITY function returns the last identity created in the same session whereas SCOPE_IDENTITY() function returns the last identity created in the same scope as well. Using SCOPE_IDENTITY() is safer.

Learn more about this rule

BP011 – NULL comparison or

addition/substring

A comparison or expression is using NULL without explicit provision for a NULL value.

Any expression or comparison that involves a NULL value will produce a NULL result. Expressions need to use IS [NOT] NULL and the ISNULL/COALESCE function to handle NULL values appropriately.

Learn more about this rule

BP012 – CASE without ELSE

The CASE statement doesn't specify what happens when all WHEN expressions evaluate to FALSE.

The ELSE clause can be omitted but unless you use the ELSE clause of the CASE statement, you will find that a NULL is returned if all WHEN expressions return FALSE. If you really want to do this, make it explicit so the code is easier to understand.

Learn more about this rule

BP013 – EXECUTE(string) is used

EXECUTE (string) is being used to execute a SQL batch in a string.

sp_ executesql   supports parameter substitution, and generates execution plans that are more likely to be reused by SQL Server.

Learn more about this rule

BP014 – [NOT] NULL option is not specified in CREATE/DECLARE TABLE statement

A column definition has not specified that a column is NULL or NOT NULL.

Because the default of allowing NULLs can be changed with the database setting 'ANSI_NULL_DFLT_ON', you should explicitly define a column as NULL or NOT NULL for noncomputed columns or, if you use a user-defined data type, that you allow the column to use the default nullability of the data type. Sparse columns must always allow NULL.

NOTE: This issue is only shown once for each table definition.

Learn more about this rule

BP015 – Scope of cursor (LOCAL/GLOBAL) is not specified

You have not explicitly defined the scope of a cursor.

When you define a cursor with the DECLARE CURSOR statement you can, and should, define the scope of the cursor name. GLOBAL means that the cursor name should be global to the connection. LOCAL specifies that the cursor name is LOCAL to the stored procedure, trigger, or batch containing the DECLARE CURSOR statement.

Learn more about this rule

BP016 – Return without result code

You have a stored procedure that does not return a result code.

When you use the EXECUTE command to execute a stored procedure, or call the stored procedure from an application, an integer is returned that can be assigned to a variable. It is generally used to communicate the success of the operation.

Learn more about this rule

BP017 – DELETE statement without WHERE clause

A DELETE statement has omitted that WHERE clause, which would delete the whole table.

It is very easy to delete an entire table when you mean to delete just one or more rows.

Learn more about this rule

BP018 – UPDATE statement without WHERE clause

The UPDATE statement has omitted the WHERE clause, which would update every row in the table.

It is very easy to delete an entire table when you mean to delete just one or more rows. Delete statements should also be in a transaction so you can check the result before committing.

Learn more about this rule

BP020 – Column created with option ANSI_PADDING set to OFF

Column created with option ANSI_PADDING set to OFF.

Learn more about this rule

BP021 – Table does not have clustered index

Most tables should have a clustered index

Learn more about this rule

BP022 – Money/SmallMoney datatype is used

The MONEY data type confuses the storage of data values with their display, though it clearly suggests, by its name, the sort of data held. Using the DECIMAL data type is almost always better.

Learn more about this rule

BP023 – Float/real datatype is used

The FLOAT (8 byte) and REAL (4 byte) data types are suitable only for specialist scientific use since they are approximate types with an enormous range (-1.79E+308 to -2.23E-308, and 2.23E-308 to 1.79E+308, in the case of FLOAT).

Any other use needs to be regarded as suspect, and a FLOAT or REAL used as a key or found in an index needs to be investigated.

The DECIMAL type is an exact data type and has an impressive range from -10^38+1 through 10^38-1. Although it requires more storage than the FLOAT or REAL types, it is generally a better choice.

Learn more about this rule

BP024 – sql_variant datatype is used

The sql_variant type is not your typical data type. It stores values from a number of different data types and is used internally by SQL Server. It is hard to imagine a valid use in a relational database. It cannot be returned to an application via ODBC except as binary data, and it isn’t supported in Microsoft Azure SQL Database.

Learn more about this rule

Didn't find what you were looking for?

Visit the Redgate forum | Contact Support

Product Articles

Tips and how-to guides for Redgate products

University

Easy to follow video courses

Community Forums

Ask, discuss, and solve questions about Redgate's tools

Events & Friends

Meet us at an event, get sponsored, and join our Friends of Redgate

Simple Talk

In-depth articles and opinion from Redgate's technical journal