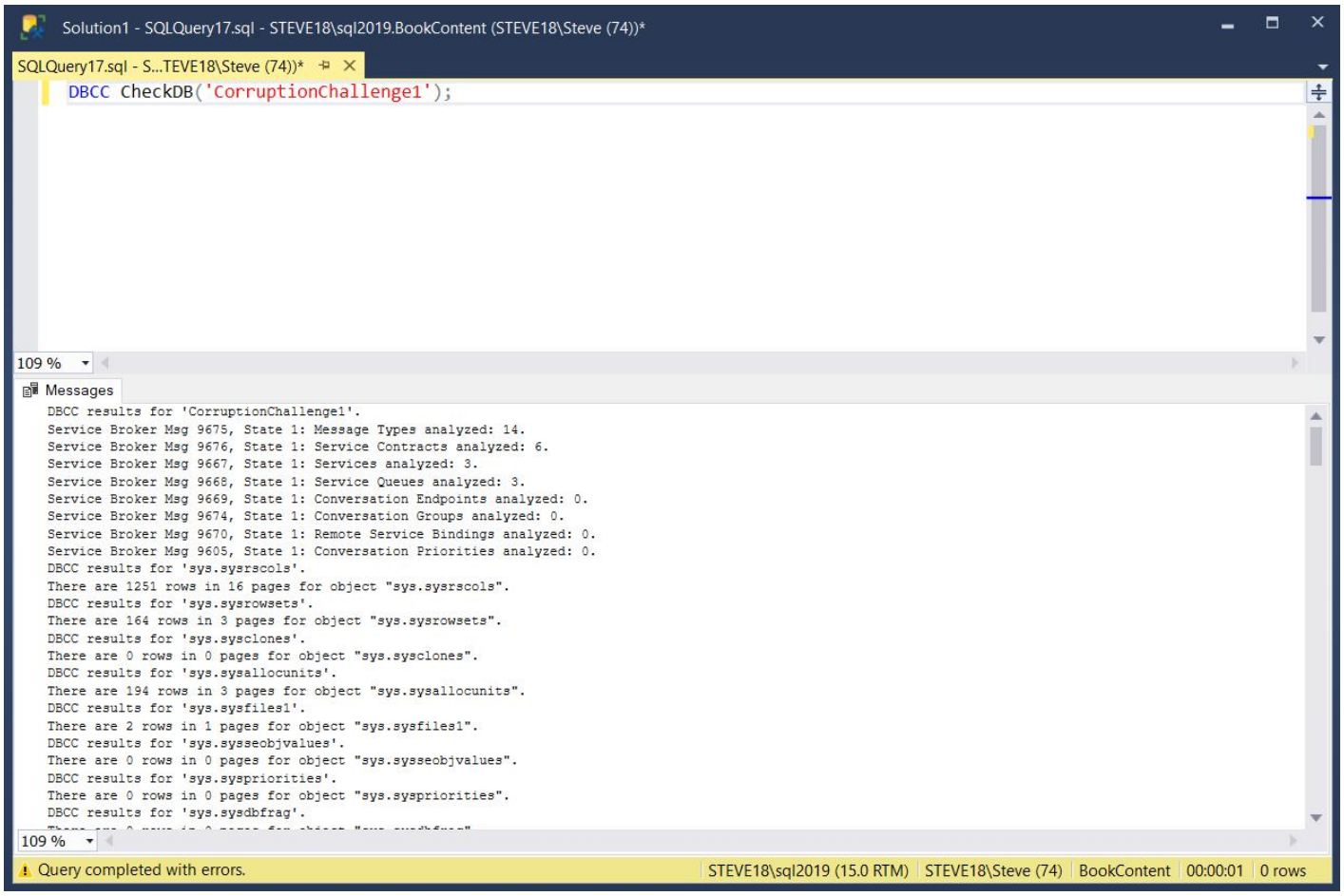
Steve Stedman
Freelance SQL Server Consultant
https://stevestedman.com

SQL Server DBCC Commands: DBCC CHECKDB

DBCC CHECKDB is a command in SQL Server that is used to check the integrity of a database. It performs a thorough analysis of all the objects in a database, including tables, indexes, and foreign keys, to ensure that they are structurally sound and do not have any corruption or inconsistencies.One of the main benefits of using DBCC CHECKDB is that it can help identify and fix issues with your database that may not be immediately apparent. For example, if there is a problem with a foreign key constraint, DBCC CHECKDB can detect this and provide recommendations for how to fix it. It can also identify problems with index fragmentation, which can cause performance issues if not addressed.To run DBCC CHECKDB on a database, you can use the following syntax:

```
DBCC CHECKDB ('database_name')
```

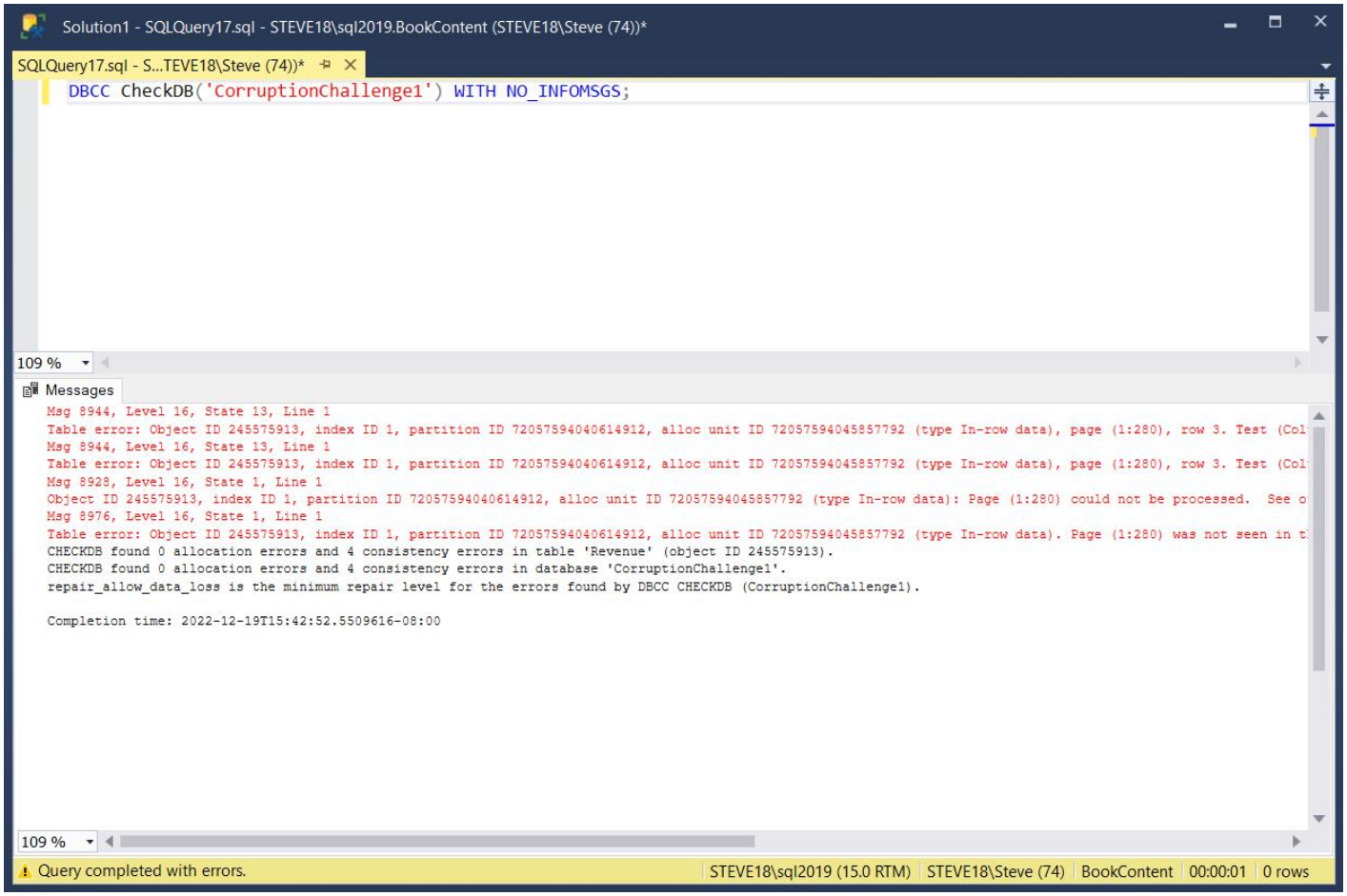


Replace database_name with the name of the database you want to check. You can also specify additional options to customize the behavior of the command.

DBCC CHECKDB is a valuable tool in SQL Server that can help ensure the integrity and consistency of your database. By regularly running DBCC CHECKDB, you can identify and fix issues that may affect the performance or reliability of your database. It is important to use caution when running this command, as it can be resource-intensive and may result in the loss of data if the REPAIR_ALLOW_DATA_LOSS option is used.

It is important to note that DBCC CHECKDB can be a resource-intensive operation, as it analyzes every object in the database. It is recommended to run DBCC CHECKDB during periods of low activity to minimize the impact on your system. You can also use the WITH TABLOCK option to lock the database and prevent any updates from occurring while the command is running.

One of the parameters that can be used with this command is NO_INFOMSGS. When this parameter is included, it suppresses the informational messages that are typically displayed when the DBCC CHECKDB command is run.



The DBCC CHECKDB command performs a number of checks on a database, including checks on the physical and logical integrity of data and the allocation structures of the database. It can also repair any minor issues that it finds. When run without the NO_INFOMSGS parameter, the

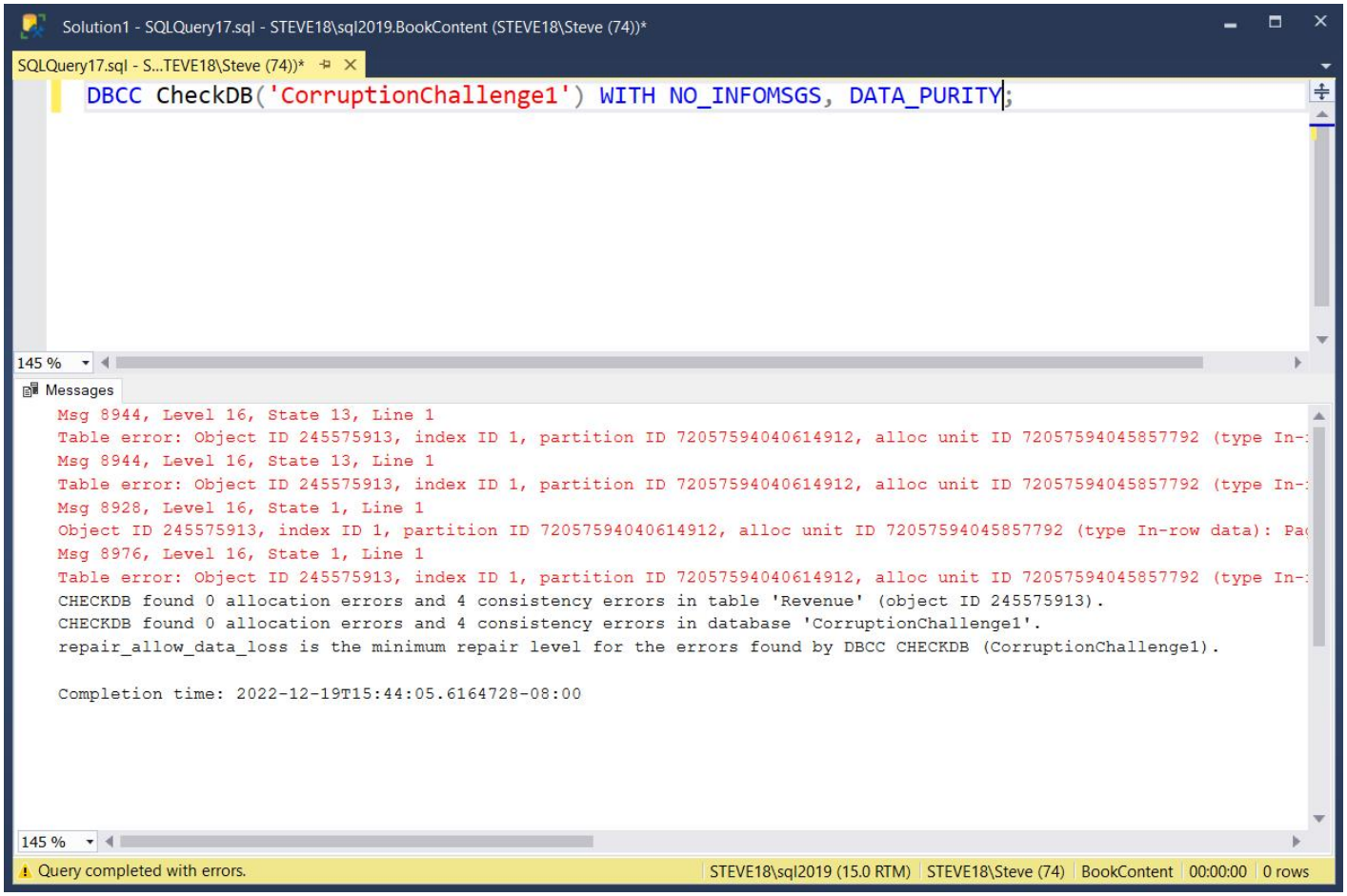
command will display a number of informational messages as it performs these checks and repairs. These messages can be helpful in understanding the status of the database and any issues that may have been found or resolved.

However, in some cases, it may be desirable to suppress these informational messages. For example, if the DBCC CHECKDB command is being run as part of an automated process or in a script, the informational messages may not be necessary or may even be disruptive. In these cases, the NO_INFOMSGS parameter can be used to suppress the messages and only display any errors that are encountered.

It's important to note that the NO_INFOMSGS parameter does not suppress all messages. It only suppresses the informational messages that are normally displayed during the DBCC CHECKDB process. Any errors or other important messages will still be displayed, even with the NO_INFOMSGS parameter in place.

NO_INFOMSGS parameter is a useful tool for suppressing the informational messages that are normally displayed when running the DBCC CHECKDB command. It can be helpful in situations where these messages are not necessary or where they may be disruptive, while still allowing important errors and other messages to be displayed.

One of the parameters that can be used with this command is DATA_PURITY. When this parameter is included, it enables the data purity checks that are performed as part of the DBCC CHECKDB process.



Data purity checks are a type of integrity check that verifies the consistency and accuracy of data in a database. These checks ensure that data adheres to the rules and constraints that have been defined for it, such as data types, foreign key relationships, and nullability. Any issues found during the data purity checks are reported as errors.

By default, data purity checks are not performed when the DBCC CHECKDB command is run. To enable these checks, the DATA_PURITY parameter must be included. This can be helpful in ensuring the quality and reliability of the data in a database, particularly if the database has undergone significant changes or has been subject to data corruption.

It's important to note that the DATA_PURITY parameter should be used with caution. While it can be helpful in identifying and correcting issues with data integrity, it can also result in a large number of errors being reported, particularly in large and complex databases. In these cases, it may be necessary to carefully review the errors and determine the appropriate actions to take to correct them.

Parameters: REPAIR_ALLOW_DATA_LOSS as a parameter to CheckDB seems like and quick and easy fix right? But do you actually know what you're doing?Well, if you read the error

messages you might see that repair_allow_data_loss is the minimum repair level for the errors found. BEWARE!This doesn't mean it will fix your data. It just means it will throw away any pages with corruption on them.If you were to run checkdb or check table with repair_allow_data_loss the SQL Server lies to you. It tells you “The error has been repaired”. With a statement like that I'd assume the corruption was repaired. But that's not the case. The repair in this case means that SQL Server threw away the page with the corruption on it. It didn't repair anything it just threw out the data. Don't make the mistake many others have! Or contact us before you run

The REPAIR_ALLOW_DATA_LOSS option for DBCC CHECKDB and DBCC CHECKTABLE can be one of the most misleading and possibly catastrophic options.

What the REPAIR_ALLOW_DATA_LOSS option does for DBCC CHECKDB and DBCC CHECKTABLE is to simply just throw away any pages (8k blocks of data) that contain rows. This may mean it is throwing away a couple of rows, or dozens to hundreds of rows when REPAIR_ALLOW_DATA_LOSS is being used.