repair_rebuild vs repair_allow_data_loss

Yesterday I posted about how repair_allow_data_loss should never be used with CHECKDB or CHECKTABLE. Today I want to share the difference between repair_rebuild vs repair_allow_data_loss.

First off, after reading the post about never using repair_allow_data_loss, and you will understand that repair_allow_data_loss simply throws away pages.

Next what repair_rebuild attempts to do is to rebuild pages in non-clustered indexes that are corrupt. I have never seen this work for clustered indexes.

Repair_rebuild – will do no harm however if doesn't help very often, but repair_allow_data_loss will throw away data.

Here is an example, using the week 1 database from the Database Corruption Challenge, however this is just showing how data is lost when you use repair_allow_data_loss with checkdb.

First we are just going to restore the Week1 database after downloading it from the Corruption Challenge.

```
RESTORE DATABASE [CorruptionChallenge1] FROM   DISK = N'C:\presentations\
DataBase Coruption Challenge\Week1\CorruptionChallenge1.bak' WITH  FILE =
 1,   MOVE N'CorruptionChallenge1' TO N'E:\SQL2017\Data\CorruptionChallen
ge1.mdf',   MOVE N'CorruptionChallenge1_log' TO N'E:\SQL2017\Logs\Corrupt
ionChallenge1_log.ldf',  NOUNLOAD, REPLACE,  STATS = 5;
```

Next we run checkDB to take a look at the problems.

```
use CorruptionChallenge1; GO  DBCC CHECKDB('CorruptionChallenge1') WITH N
O_INFOMSGS;
```

We can see the corruption in the Revenue table, so we take a look at how many rows we have and how much revenue has been recorded in that table.

```
SELECT COUNT(*), sum(Revenue) as TotalReveune  FROM Revenue;
```

You can see, 54 rows in the table with $16,283,040 recorded in revenue. Imagine if this was the revenue for your department, not bad depending on what you might be selling.

Next we try checkdb with repair_rebuild.

```
use CorruptionChallenge1; go ALTER DATABASE CorruptionChallenge1 SET SING
LE_USER WITH ROLLBACK IMMEDIATE; DBCC CHECKDB('CorruptionChallenge1', REP
AIR_REBUILD) ; ALTER DATABASE CorruptionChallenge1 SET MULTI_USER;   DBCC
 CHECKDB('CorruptionChallenge1') WITH NO_INFOMSGS; SELECT COUNT(*), sum(R
evenue) as TotalReveune FROM Revenue;
```

The repair_rebuild does nothing in this case, and our revenue table remains unchanged.

Then we try checkdb with repair allow data loss.

```
use CorruptionChallenge1; go ALTER DATABASE CorruptionChallenge1 SET SING
LE_USER WITH ROLLBACK IMMEDIATE; DBCC CHECKDB('CorruptionChallenge1', REP
AIR_ALLOW_DATA_LOSS) ; ALTER DATABASE CorruptionChallenge1 SET MULTI_USER
;  DBCC CHECKDB('CorruptionChallenge1') WITH NO_INFOMSGS;  SELECT COUNT(*
), sum(Revenue) as TotalReveune FROM Revenue;
```

Which says that it repaired things, but the output lies. (Lies shown with the blue arrows).

Technically it didn't repair anything, instead it just threw away data. We can see that when we check the revenue table again to see how many rows are there and how much revenue we have. Yes the corruption has been removed, but so has our data.

```
SELECT COUNT(*), sum(Revenue) as TotalReveune  FROM Revenue;
```

Somehow we are missing $10,901,260 in revenue. Try to explain that one to your CFO. CheckDB "fixed" the database by throwing away half of the rows in this table.

You can run checkdb with repair_allow_data_loss in a transaction to see what is going to happen then to run it back. This is safe to do in most scenarios.

```
use CorruptionChallenge1; go ALTER DATABASE CorruptionChallenge1 SET SING
LE_USER WITH ROLLBACK IMMEDIATE; BEGIN TRANSACTION   DBCC CHECKDB('Corrup
tionChallenge1', REPAIR_ALLOW_DATA_LOSS) ;   DBCC CHECKDB('CorruptionChal
lenge1') WITH NO_INFOMSGS; ROLLBACK TRANSACTION; ALTER DATABASE Corruptio
nChallenge1 SET MULTI_USER;
```

I still stick with my statement of never running repair allow data loss without knowing what is going to be lost, and without a way to get it back. Also most any time that you have a scenario that repair_rebuild will work, it is usually just as easy to fix an index by dropping and recreating it.

Good luck with your corruption repair, and if you need any help, please reach out to us at Stedman Solutions before ever running REPAIR_ALLOW_DATA_LOSS.

-Steve Stedman