Corruption Challenge 4 - Alternate Solution

Here is the solution provided by Patrick Flynn. Congratulations Patrick for being one of this weeks winners.

I really liked this solution for a number of reasons, first it showed how to pull data from DBCC Page and actually convert it into rows and columns that could then be inserted back into a table. Secondly due to its use of Common Table Expressions.

This was done using SQL Server 2014.

Approach used was:

(a) Restore and check database – Corruption is related to Allocation Meta Data for dbo.Customers

(b) Use the non-clustered indexes on FirstName and LastName to recover data apart from Middle name – Total of 511740 rows in Table
Steps are in CorruptionChallange Setup.sql (requires disabling Database triggers

(c) Cannot directly select data from pages due to allocation issues but can select raw data vis DBCC Page (and DBCC IND)
Requires using DBCC Ind to get list of Data pages and DBCC Page to extract data
Steps to extract and Parse data are in DBCCPage_Extract

(d) After checking data was match by Id, FirstName and Last Name restore data into Customer table
This requires temporarily disabling FK on Orders table

(e) Check Database Corruption fixed

(f) Cleanup objects created and re-enable FK's and Triggers
Steps are in CorruptionChallange_Cleanup.sql

Start by restoring the database. Of course Patrick was using different paths, so I had to modify the file paths to work on my test server.

Step 1 restore the database.

Step2 DBCC CheckDB().
The next thing to try is DBCC CheckDB, however when I run this, it eventually crashes out and drops the connection to the SQL Server from SSMS.

If DBCC CheckDB is run, and then cancelled after 6 seconds, it already has returned 78,000 lines of output ending with:

```
Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 249. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 250. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 251. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 252. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 253. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 254. Column 'subid' was created NOT NULL, but is NULL in the row.
 Msg 8970, Level 16, State 1, Line 1
 Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 255. Column 'subid' was created NOT NULL, but is NULL in the
```

row.
Msg 8970, Level 16, State 1, Line 1
Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 256. Column 'subid' was created NOT NULL, but is NULL in the row.
Msg 8970, Level 16, State 1, Line 1
Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 257. Column 'subid' was created NOT NULL, but is NULL in the row.
Msg 8970, Level 16, State 1, Line 1
Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 258. Column 'subid' was created NOT NULL, but is NULL in the row.
Msg 8970, Level 16, State 1, Line 1
Row error: Object ID 29, index ID 1, partition ID 281474978611200, alloc unit ID 1900544 (type In-row data), page ID (3:192), row ID 259. Column 'subid' was created NOT NULL, but is NULL in the row.
CHECKDB found 0 allocation errors and 39344 consistency errors in table 'sys.sysprivs' (object ID 29).
CHECKDB found 0 allocation errors and 39344 consistency errors in database 'CorruptionChallenge4'.
repair_allow_data_loss is the minimum repair level for the errors found by DBCC CHECKDB (CorruptionChallenge4).
Query was cancelled by user.

Now on to do something about that corruption.
Step 3 Check Current User Objects - Before == 5

Step 4 Errors are related to Clustered Index
Can use the Non clustered indexes to recover all data except MiddleName
Require to disable Database Triggers

-- GET what we can from the indexes.

511740 rows recovered

Shows that we get back the data, but no middle names.

```
-- Using DBCC Page data is intact in page but Allocation Data is incorrect
-- Can extract data via DBCC Page and Parse Data to obtain MiddleName
-- Create Table to store data from DBCC Page
```

Now create some temp tables, and use a cursor to fill them up using DBCC page.

So now all the output from DBCC Page has been stored in the #DBCC_Page table. Let’s take a look at what’s there.

Now to translate the data and store it into a table called tmpCustomerData.

Let’s take a look at the contents of the [tmpCustomerData] table.

The next step requires this function HexStrToVarBinary, so we will create it now.

Then using our new function, and another Common Table Expression, the data is extracted and placed into a table called [Customer_Copy1].

Check the contents of the [Customers_Copy1] table.

Compare to data extracted from Non-Clustered, and they all match on the id.

-- 511740 rows matching
Now to see how many names don’t match.

Time to clean up and put all the data back into the original table.
Cleanup Step1 Restore Data to original Table (need to disableFK)

Now confirm that the corruption is fixed.

Put back everything we removed.

And finally the check needed to submit the challenge this week.

And that’s how it’s done if you want to rip through the data pages with DBCC Page. Nice work Patrick.