DECODING KEY AND PAGE WAITRESOURCE FOR DEADLOCKS AND BLOCKING

ON OCTOBER 17, 2016

In this post...

- Page lock waits
- Key lock waits
- Credits and more reading

If you use SQL Server's blocked process report or collect deadlock graphs, occasionally you'll come across things that look like this:

> waitresource="PAGE: 6:3:70133 " waitresource="KEY: 6:72057594041991168 (ce52f92a058c)"

Sometimes there's more information in the massive monster of XML that you're scanning through (deadlock graphs have a resource list that help reveal the object and index name), but sometimes there isn't.

Here's a reference on how to decode them.

All of this information is out there on the internet already in various places, it's just spread out! I'm going to pull the whole thing together, from DBCC PAGE to hobt_id to the undocumented %%physloc%% and %%lockres%% functions.

First we'll talk through PAGE lock waits, then we'll hit the KEY lock waits.

Page lock waits

Example 1: waitresource=“PAGE: 6:3:70133 " = Database_Id : FileId : PageNumber

If your query was waiting on a page level lock, SQL Server gives you the page address.

Breaking “PAGE: 6:3:70133” down, we’ve got:

- database_id=6
- data_file_id = 3
- page_number = 70133

1.1) Decode the database_id

Find the database name with this query:

```
SELECT
    name
FROM sys.databases
WHERE database_id=6;
GO
```

That’s the WideWorldImporters sample database (/2016/09/13/deadlock-code-for-the-wideworldimporters-sample-database/) on my SQL Server instance.

1.2) Look up the data file name – if you’re interested

We’re going to use the data file id in the next step to find the name of the table. You can just move on. But if you’re curious about the name of the data file, you can look it up by using the database and plugging the data file id into this query:

```
USE WideWorldImporters;
GO
SELECT
    name,
    physical_name
FROM sys.database_files
WHERE file_id = 3;
GO
```

In WideWorldImporters, this is the data file named WWI_UserData, and I restored it to C:\MSSQL\DATA\WideWorldImporters_UserData.ndf. (Whoops, you caught me putting files on my system drive.).

1.3) Get the name of the object from DBCC PAGE

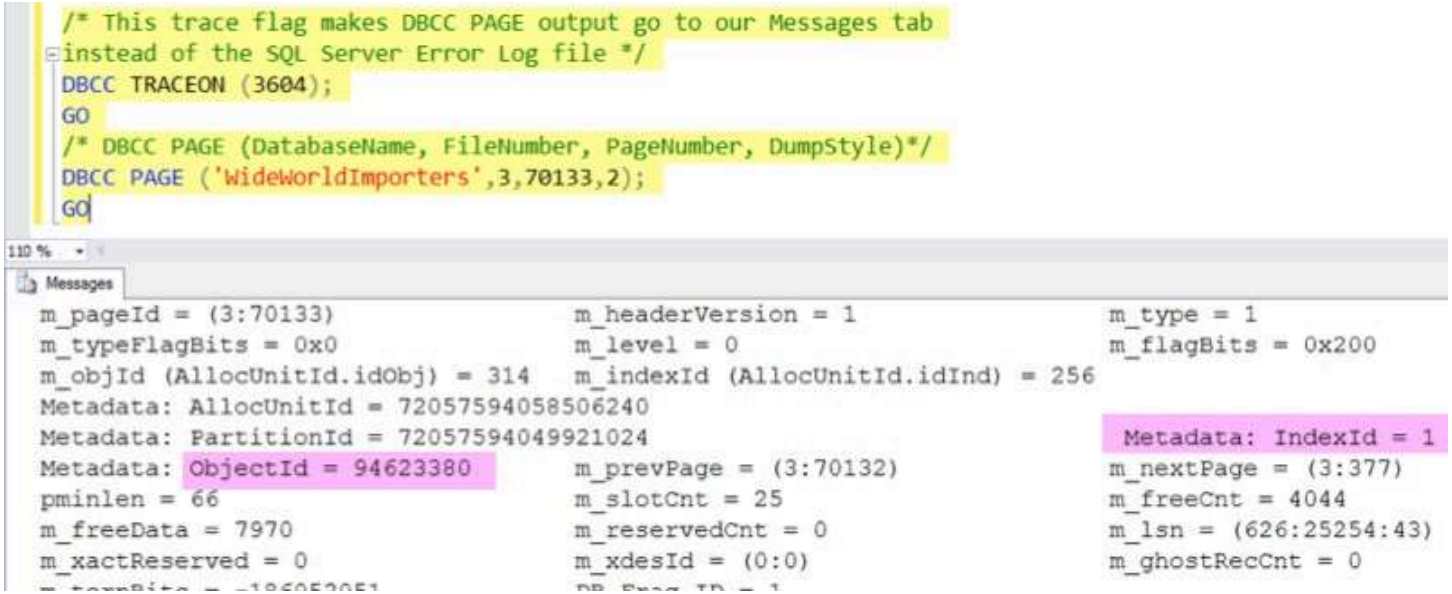
We know this is page # 70133 in data file 3 in the WideWorldImporters database. We can look at that page with the undocumented DBCC PAGE and Trace Flag 3604.

Note: I prefer running DBCC page against a restored backup elsewhere, because it's not supported. In some cases, running DBCC PAGE can cause stack dumps (https://connect.microsoft.com/SQLServer/feedback/details/776144/dbcc-page-incorrect-output-with-filtered-indexes).

```
/* This trace flag makes DBCC PAGE output go to our Messages tab
instead of the SQL Server Error Log file */
DBCC TRACEON (3604);
GO

/* DBCC PAGE (DatabaseName, FileNumber, PageNumber, DumpStyle)*/
DBCC PAGE ('WideWorldImporters',3,70133,2);
GO
```

Scrolling down in the output, I can find the object_id and IndexId:



Whew, almost there!

I can now find the table and index name with this query:

```
USE WideWorldImporters;
GO
SELECT
    sc.name as schema_name,
    so.name as object_name,
    si.name as index_name
FROM sys.objects as so
JOIN sys.indexes as si on
    so.object_id=si.object_id
JOIN sys.schemas AS sc on
    so.schema_id=sc.schema_id
WHERE
    so.object_id = 94623380
    and si.index_id = 1;
GO
```

And behold, this lock wait was on the PK_Sales_OrderLines index on the Sales.OrderLines table.

Note: In SQL Server 2014 and higher, you could also find the object name using the undocumented sys.dm_db_database_page_allocations dynamic management object. But you have to query all the pages in the database, which seems not as awesome against large databases – so I listed the DBCC page method.

1.4) Can I see the data on the page that was locked?

Well, yes. But … do you really need to?

This is slow even on small tables. But it's kinda fun, so… since you read this far… let's talk about %%physloc%%!

%%physloc%% is an undocumented piece of magic that will return the physical record locator for every row. You can use %%physloc%% with sys.fn_PhysLocFormatter in SQL Server 2008 (http://www.sqlskills.com/blogs/paul/sql-server-2008-new-undocumented-physical-row-locator-function/) and higher.

Now that we know that the page lock wait was on Sales.OrderLines, we can see all the data in that table on data file = 3 and page number = 70133 with this query:

```
Use WideWorldImporters;
GO
SELECT
    sys.fn_PhysLocFormatter (%%physloc%%),
    *
FROM Sales.OrderLines (NOLOCK)
WHERE sys.fn_PhysLocFormatter (%%physloc%%) like '(3:70133%'
GO
```

Like I said, this is slow even on tiny tables. I've added NOLOCK to the query because while we want a glance at this info, we have no guarantee that it's the way it was earlier when the blocking happened anyway– we're guessing, so we may as well do dirty reads.

But woo hoo, it gives me a clean display of the 25 rows which the query was fighting for:

```
Use WideWorldImporters;
GO
SELECT
    sys.fn_PhysLocFormatter (%%physloc%%),
    *
FROM Sales.OrderLines
WHERE sys.fn_PhysLocFormatter (%%physloc%%) like '(3:70133%'
GO
```

110 %

Results | Messages

	(No column name)	OrderLineID	OrderID	StockItemID	Description	Pac
1	(3:70133:0)	231365	73574	19	DBA joke mug - daaaaaa-ta (Black)	7
2	(3:70133:1)	231366	73574	191	Black and orange fragile despatch tape 48mmx75m	7
3	(3:70133:2)	231367	73575	40	Developer joke mug - (hip, hip, array) (White)	7
4	(3:70133:3)	231368	73575	200	Black and yellow heavy despatch tape 48mmx100m	7
5	(3:70133:4)	231369	73575	126	Plush shark slippers (Gray) S	7
6	(3:70133:5)	231370	73575	35	Developer joke mug - fun was unexpected at this ti...	7

That's enough detail on waitresource=PAGE. What if you were waiting on a KEY?

Key lock waits

Example 2: waitresource="KEY: 6:72057594041991168 (ce52f92a058c)" = Database_Id, HOBT_Id (Magic hash that you can decode with %%lockres%% if you really want)

If your query was trying to lock a row in an an index and was blocked, you get a totally different style of address.

Breaking "6:72057594041991168 (ce52f92a058c)" down, we've got:

- database_id = 6
- hobt_id = 72057594041991168
- magic hash value = (ce52f92a058c)

2.1) Decode the database_id

This works the exact same way it did for the page example above! Find the database name with this query:

```
SELECT
    name
FROM sys.databases
WHERE database_id=6;
GO
```

In my case, that's still the WideWorldImporters sample database (/2016/09/13/deadlock-code-for-the-wideworldimporters-sample-database/).

2.2) Decode the hobt_id

We need to use that database, and then query sys.partitions, with some helper joins to figure out the table and index name…

```
USE WideWorldImporters;
GO
SELECT
    sc.name as schema_name,
    so.name as object_name,
    si.name as index_name
FROM sys.partitions AS p
JOIN sys.objects as so on
    p.object_id=so.object_id
JOIN sys.indexes as si on
    p.index_id=si.index_id and
    p.object_id=si.object_id
JOIN sys.schemas AS sc on
    so.schema_id=sc.schema_id
WHERE hobt_id = 72057594041991168;
GO
```

That tells me that the query was waiting for a lock on Application.Countries, using the PK_Application_Countries index.

2.3) Now for some %%lockres%% magic - if you want to figure out which row was locked

If I really want to know exactly which row the lock needed, I can decode that by querying the table itself. We can use the undocumented %%lockres%% function to find the row equal to that magic hash value.

Note that this is going to scan the table, and on large tables that might not be so awesome all the time:

```
SELECT
    *
FROM Application.Countries (NOLOCK)
WHERE %%lockres%% = '(ce52f92a058c)';
GO
```

I added NOLOCK to this query (as Klaus Aschenbrenner suggested on Twitter (https://twitter.com/Aschenbrenner/status/788036874891853824)) because locking can be an issue – and in this case, you're looking to get a glance at the data as it is now, not as it was earlier when the transaction ran– so I don't think data consistency is a big issue.

Voila, the row we were fighting for appears!

Credits and more reading

I'm not sure who first documented many of these things, but here are two posts on some of the less documented nitty gritty that you may enjoy:

- Paul Randal's post on %%physloc%% and sys.fn_PhysLocFormatter (http://www.sqlskills.com/blogs/paul/sql-server-2008-new-undocumented-physical-row-locator-function/) (how we saw the data on the page lock wait example)
- This StackOverflow question on using %%lockres%% (http://dba.stackexchange.com/questions/106762/how-can-i-convert-a-key-in-a-sql-server-deadlock-report-to-the-value) (how we found the row on the row lock wait example). One of the answers links to Grant Fritchey writing about %%lockres%% back in 2010 (http://www.scarydba.com/2010/03/18/undocumented-virtual-column-lockres/).

SQLCOMICS (/sqlcomics/)

ABOUT (/about/)

PODCAST (/dear-sql-dba-podcast/)

EVENTS (/events/)

FREE COURSES (/coursesbytitle/)

YOUTUBE (https://www.youtube.com/channel/UCrJ8WLrVoKxL94mKv2akxTA)

TIKTOK (https://www.tiktok.com/@kendra_little)

BLOG RSS FEED (/blog/rss.xml)

SQLCOMICS RSS FEED (/sqlcomics/rss.xml)

★ REGISTER (https://passdatacommunitysummit.com/sessions/?category=Pre-conference&level=200&track=Database+Management&type=Community)

SEARCH

Search

POSTS OR COMICS FOR YOUR INBOX?

Email Address

SUBSCRIBE

PREV « WHERE DO YOU GET YOUR CREATIVITY? (DEAR SQL DBA EPISODE 20) (/2016/10/20/WHERE-DO-YOU-GET-YOUR-CREATIVITY-DEAR-SQL-DBA-EPISODE-20/)

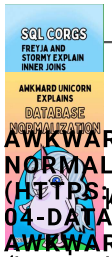
NEXT » SHOULD YOU REBUILD OR REORGANIZE INDEXES ON LARGE TABLES? (DEAR SQL DBA EPISODE 19) (/2016/10/13/SHOULD-YOU-REBUILD-OR-REORGANIZE-INDEXES-ON-LARGE-TABLES-DEAR-SQL-DBA-EPISODE-19/)

LATEST SQLCOMICS

SQL CORGS EXPLAIN INNER JOINS (HTTPS://KENDRALITTLE.COM/SQLCOMICS/2024-04-SQL-CORGS-EXPLAIN-INNER-JOINS/)

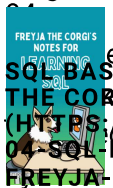
RECENT POSTS

SQL SERVER PAGE COMPRESSION: SHOULD YOU WORRY ABOUT CPU USAGE INCREASING ON ON



AWKWARD UNICORN EXPLAINS DATABASE NORMALIZATION

(HTTPS://KENDRALITTLE.COM/SQLCOMICS/2024-04-DATABASE-NORMALIZATION-EXPLAINED-BY-AWKWARD-UNICORN/)



SQL BASICS (SELECT, FROM, WHERE) WITH FREYJA THE CORGI

(HTTPS://KENDRALITTLE.COM/SQLCOMICS/2024-04-SQL-BASICS-SELECT-FROM-WHERE-WITH-FREYJA-THE-CORGI/)

WHERE AM I?

Kendra Little's blog on SQL Server with SQL Comics.

GO TO CONTACT PAGE (/CONTACT)

SUBSCRIBE

Email Address

SUBSCRIBE

INSERTS, UPDATES, AND DELETES?

(HTTPS://KENDRALITTLE.COM/2024/09/08/SQL-SERVER-PAGE-COMPRESSION-CPU-ON-INSERT-UPDATE-DELETE/)

FREE SCRIPT TO AUTOMATE UNFORCING FAILED FORCED PLANS IN QUERY STORE (SQL SERVER)

(HTTPS://KENDRALITTLE.COM/2024/09/02/FREE-SCRIPT-AUTOMATE-UNFORCING-FAILED-FORCED-PLANS-QUERY-STORE-SQL-SERVER/)

PLEASE COMPRESS YOUR INDEXES AND SHRINK YOUR DATABASES FOR AZURE SQL MANAGED INSTANCE

(HTTPS://KENDRALITTLE.COM/2024/08/26/PLEASE-COMPRESS-INDEXES-AND-SHRINK-YOUR-DATABASE-AZURE-SQL-MANAGED-INSTANCE/)