



10 Minute Triage: Troubleshooting Production Issues 101

Michael Krafiick (Twitter: @mkrafiick)
Lead Db2 Database Engineer, Atlanta

Session Code: H6
Thursday, September 13, 2018 (9:45 am) | Platform: LUW



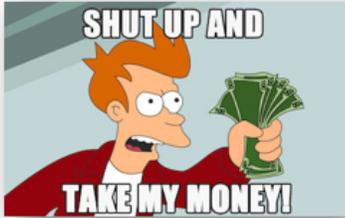
Session Objectives

- Why is speed of recovery important? (Cost of database downtime)
- “What is Normal?”
- Inefficiency vs. Pain. How does one lead to another?
- Common bottlenecks and unexpected ripple affects.
- Developing an assessment plan and a triage toolbox.

The Cost of Downtime

Hey Mike, can you remember \$237.00 a minute?

Wrap Your Head Around the Scope



Downtime	Cost in Penalties	Could Buy
15 Minutes	\$3,555.00	1-2 Macbook Pros
30 Minutes	\$7,110.00	1 Mo "Avg" I/T Salary
45 Minutes	\$10,665.00	1-2 High End Big Screen TVs
60 Minutes	\$14,220.00	Used Car
4 Hours	\$56,880.00	1-2 at a University
8 Hours	\$113,760.00	Tesla Roadster
12 Hours	\$170,640.00	143,396 Tacos at Taco Bell
16 Hours	\$227,520.00	Cabin in the Mountains
24 Hours	\$341,280.00	More than my house

What is Normal?

Not the client currently running around with their hair on fire.

or

It's not my circus, not my monkey.

You Can't Manage What You Don't Measure

Information is important for capacity planning and trending, but it is critical in a disaster.

What is Normal?

Is a .5 second query execution in a online banking application normal?

How about a 25 minute pipeline report execution in a Data Warehouse?

Holy cow, we have 750 connected users at 10am!

Should my data disk be spinning at 4000 IOPS?

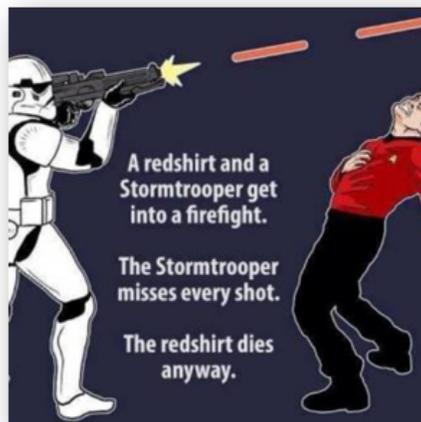
Defining “Normal”

Collect metrics that serve a purpose. Spearfish data and avoid the fishing net.

- Metric collection has an impact. What footprint will you leave?
- Characteristics of good data collection:
 1. Easy to collect
 2. Easy to understand
 3. Relevant
 4. Sensitive, without false alarm. A variance in metric that signals change

Characteristics of good data collection used with permission - "Tuning and Monitoring Database System Performance" by Steve Reese (pg.18)

Be a SEAL Team, not a Storm Trooper



Sniping the Metrics

- Application Level - Key Performance Indicators (KPI)
Abandoned Shopping Carts | Website Traffic | Order Fulfillment
- Database Level
 - Performance (How have we been? How are we doing?)
 - Pain (Early warning system.)
 - Configuration (How should we be running?)
- Operating System / Server Level
CPU Saturation | IO Per Second (Disk) | Swap Space (Memory)

Efficiency vs. Pain

Man, I didn't see that coming.

Why do I differentiate between the two?

Because one shows we are broken, another shows how we got to the broken state.

So monitor on all, but alert on few.

Efficiency Metrics Show Stress

Typically shows a slow build of problems or capacity issues coming.

Efficiency:

<i>Index Read Efficiency</i>	Top 10 SQL – Number Executions
<i>Avg. Result Set Size</i>	Top 10 SQL – Execution Time
Lock Escalation / Lock Waits / Deadlocks	CPU Utilization
Bufferpool Hit Ratio	I/O Wait Time
Database Inside vs. Outside Time	

These are metrics I traditionally try to record for a historical period of time. Allows for capacity planning, shows behavior changes after a release, and allows you to go back and troubleshoot after a problem starts.

A Unique Efficiency Metric

Poor efficiency is when you "bend", pain is when you "break" .

Index Read Efficiency (IREF) = Rows Read / Rows Returned



ARSS (at a database level): `ROWS_RETURNED/SELECT_SQL_STMTS`

Script to Determine ARSS and IREF at the DISCOVER.DB2 repository of my GitHub Account:

DiscoverDB2/IREF.sql -

<https://github.com/MKrafcik/DiscoverDB2/blob/master/IREF.sql>

Flushing Out Inefficiency with Brother-Panther

Follow Up	Stmt ID	Verb	Type	# Execs	% CPU Time	% Exec Time	Avg Exec Time (sec)	% Data L Reads	Avg L Reads	% Temp Data L Reads	Avg Temp L Reads	% Rows Read	Avg Rows Read	IX Read Efficiency	% Rows Fetched	Avg Rows Fetched
	99174801...	UPDATE	DYNAMIC	726	44.104%	28.665%	0.371287	48.178%	270,558.624	0.000%	0.000	48.567%	261,470.837	261,470.837	0.000%	0.000
	D85524462...	SELECT	DYNAMIC	733	33.656%	21.905%	0.281019	50.180%	274,734.662	3.176%	3.023	50.650%	270,077.895	1,473.660	12.194%	183.270
	D488C148D...	DELETE	DYNAMIC	724	0.968%	0.510%	0.004630	0.414%	5,083.572	0.000%	0.000	0.372%	2,008.128	2,008.128	0.000%	0.000
	D01100960...	SELECT	DYNAMIC	1	0.000%	0.000%	0.002650	0.000%	237.000	0.000%	0.000	0.000%	89.000	1.000	0.000%	89.000
	FFF1A020...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000179	0.000%	4.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FFE604E9...	SELECT	DYNAMIC	1,744	0.024%	0.011%	0.000059	0.000%	1.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FF80C21C...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000164	0.000%	1.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FE055179...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000107	0.000%	1.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FB504497...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000142	0.000%	4.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	F8A51468...	SELECT	DYNAMIC	1	0.000%	0.001%	0.004828	0.000%	156.000	0.000%	0.000	0.000%	62.000	1.000	0.006%	62.000
	F9A5177D...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000096	0.000%	1.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	F8E99302...	SELECT	DYNAMIC	6,539	0.126%	0.056%	0.000081	0.006%	6.880	0.000%	0.000	0.006%	3.851	1.000	2.286%	3.851

Index Read Efficiency (IREF)

< 10	Great
10 - 100	Possible
100 - 1000	Poor
1000+	Severe



Notice that in this 7 hour period, one sub-second SQL executed so much that it consumed 44% of the CPU during that time period.

Flushing Out Inefficiency with Brother-Panther

The screenshot shows the Brother-Panther interface with the following content:

Analysis Report Output

```

Trying variations of the solution set.
  1 indexes in current solution
[ 13.5407] timerons (without recommendations)
[  0.0062] timerons (with current solution)
[99.95%] improvement
  
```

Analysis Messages

```

Operating System = [redacted] Feb 10 11:10:31 est 2
Execution Method = DBI
Advisor Command = db2advia -t 0 -p [redacted] -w DBI_Analysis_Wor
Creating Explain Plan Tables...Done
Using user id as default schema name. Use -n option to specify schema
  
```

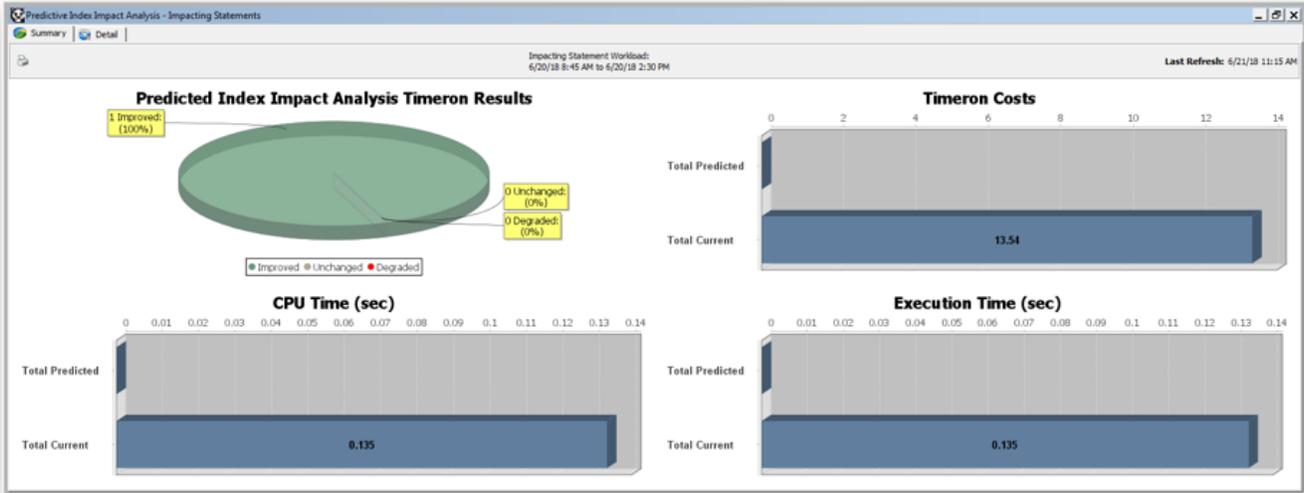
Recommended Indexes

Table Schema	Table Name	Index Name	Index Size	Add Benefi...	Subtract Loss %	Index Columns
CMGLSR	YFS_INVENTORY_ACTIVITY	IDX1806211501520	0.00MB	99.95%	100.00%	INVENTORY_ITEM_KEY
CMGLSR	YFS_INVENTORY_ACTIVITY	INV_ACTY_I2	0.00MB	N/A	N/A	PROCESSED_FLAG-NEXT_PROCESSOR

A red circle highlights the context menu for the selected row, showing options: Properties, Impact Analysis (with a sub-menu containing Run Impact Analysis and Change Timeframe), Create, Select All, and Copy.

Normally a cost of 13 Timerons wouldn't make a DBA even blink with concern, but the IREF shows it as a red flag and Brother-Panther can show how it consumes CPU.

Flushing Out Inefficiency with Brother-Panther



Flushing Out Inefficiency with Brother-Panther

Brother-Panther®

Statement Performance for [redacted]

Statement Workload from 7/6/18 10:00 AM to 7/6/18 1:00 PM

Last Refresh: 7/6/18 2:07 PM

Follow Up	Stmnt ID	Verb	Type	# Execs	% CPU Time	% Exec Time	Avg Exec Time (sec)	% Data I Reads	Avg L Reads	% Temp Data I Reads	Avg Temp L Reads	% Rows Read	Avg Rows Read	IX Read Efficiency	% Rows Fetched	Avg Rows Fetched
	99174801...	UPDATE	DYNAMIC	3,328	2.859%	0.672%	0.000275	3.112%	86,648	0.000%	0.000	0.000%	2,409	2,409	0.000%	0.000
	D852A462...	SELECT	DYNAMIC	3,327	0.971%	2.243%	0.000099	1.555%	4,425	0.000%	0.000	0.000%	2,408	1.000%	2,359%	2,408
	D48EC18D...	DELETE	DYNAMIC	3,322	1.725%	0.406%	0.000166	1.979%	59,809	0.000%	0.000	0.000%	0.000	0.000%	0.000%	0.000
	032F3A56...	SELECT	DYNAMIC	25	0.016%	0.004%	0.000020	0.000%	1,000	0.000%	0.000	0.000%	0.000	0.000%	0.000%	0.000
	FF2404E9...	SELECT	DYNAMIC	924	0.156%	0.041%	0.000060	0.000%	1,000	0.000%	0.000	0.000%	0.000	0.000%	0.000%	0.000
	F8444E4E...	SELECT	DYNAMIC	17	0.061%	0.016%	0.001253	0.023%	191,765	0.000%	0.000	0.005%	7,000	1.000%	0.035%	7,000

Brother-Panther®

Compare Statement Workload

Before: 6/20/18 8:00 AM to 6/20/18 3:00 PM

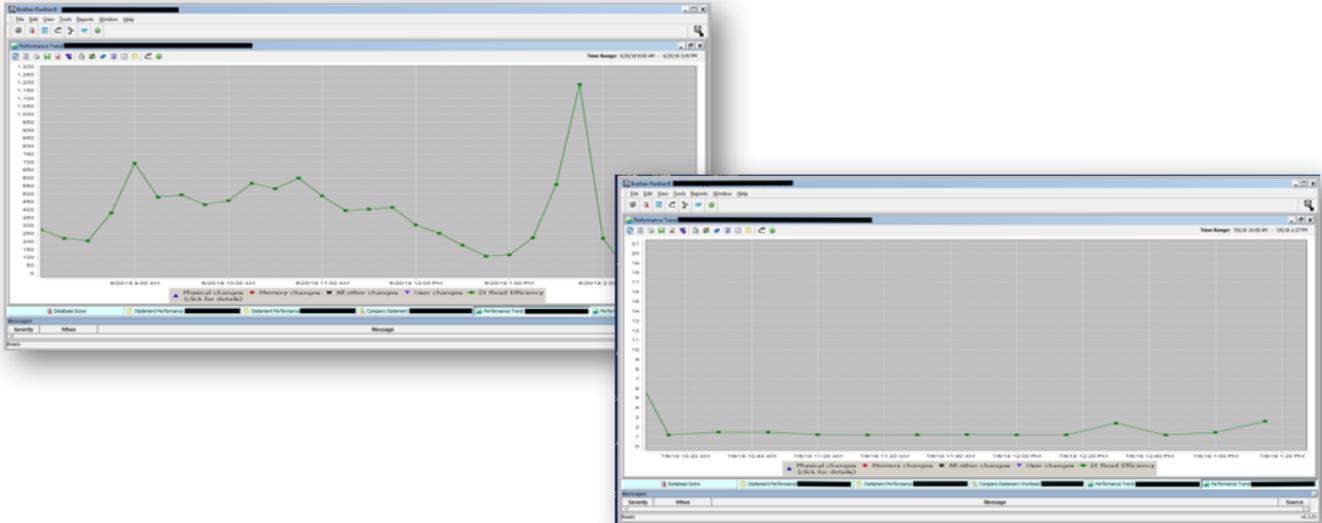
After: 7/6/18 10:00 AM to 7/6/18 1:00 PM

Last Refresh: 7/6/18 2:11 PM

Rows: 649 Max Rows: 5000

Follow Up	Stmnt ID	Avg Exec Time (s) Improved	Avg Exec Time (s) Before	Avg Exec Time (s) After	Avg Exec Time (s) Change	% Exec Time After	IX Read Efficiency Improved	IX Read Efficiency Before	IX Read Efficiency After	IX Read Efficiency Change	Avg IX L Reads Improved	Avg IX L Reads Before	Avg IX L Reads After	Avg IX L Reads Change	% IX L Reads After	Avg Data L Reads Improved
	D852A462...	Y	0.281019	0.000099	-0.280920	0.243%	Y	1,473,660	1,000	-1,472,660	Y	4,810,563	2,017	-4,808,546	0.202%	Y
	D48EC18D...	Y	0.006630	0.000166	-0.006464	0.406%	Y	2,808,128	0.000	-2,808,128	Y	3,826,544	55,741	-2,770,803	5.576%	Y
	99174801...	Y	0.371287	0.000275	-0.371012	0.672%	Y	261,470,837	2,409	-261,468,428	Y	8,904,079	81,831	-8,822,248	8.201%	Y
	0011D090...		0.002490	0.000000	0.000000	0.000%		1,000	0.000	0.000		138,000	0.000	0.000	0.000%	
	FF2404E9...		0.000179	0.000000	0.000000	0.000%		0.000	0.000	0.000		4,000	0.000	0.000	0.000%	
	FF80C21C...		0.000164	0.000000	0.000000	0.000%		0.000	0.000	0.000		1,000	0.000	0.000	0.000%	

Flushing Out Inefficiency with Brother-Panther





DISCOVER
Db2

Finding
Performance Hogs

www.youtube.com/DiscoverDB2



DISCOVER.DB2

Published on Jun 19, 2018

There are tons of metrics to show pain, but there are just as many to show efficiency. Index Read Efficiency (IREF) is a way to find SQL performance hogs that you may not normally catch. In today's episode, we will talk about the metric and how to interpret it.

Timestamps:

- 01:03 - What is Index Read Efficiency (IREF)
- 02:12 - IREF Thresholds
- 03:13 - IREF in context with Workload Type (ARSS)
- 05:32 - Analyzing Example Output
- 08:30 - SQL for find ARSS and IREF
- 10:13 - Github Repository with SQL

Useful Links:

- (Blog) Boost Your Performance - <https://tinyurl.com/ydyoaats>
- (DBI/Blog) Advice: IREF - <https://tinyurl.com/ybwadpuq>
- (DBI/Blog) Performance: ARSS - <https://tinyurl.com/ycxa4cfu>
- (Xtivia/Blog) IREF, a KPI - <https://tinyurl.com/yczv2qag>
- (Blog) What is a Table Scan - <https://tinyurl.com/ya2w9q5v>
- DBI's Brother Panther - <https://tinyurl.com/yb22k763>
- Xtivia - <http://www.xtivia.com/>

YouTube Channel: <https://www.youtube.com/DiscoverDB2>

Pain Metrics Act as an Early Warning System

Early warning indicators, actively triage for a problem.

Pain:

Database Up/Down	HADR Congestion / Role Change
Lock Escalation / Lock Waits / Deadlocks	CPU Utilization
Number <i>Active</i> Connections	Disk Utilization
Active/Archive Log Utilization	Swap Space Utilization
Tablespace State	
Database Inside vs. Outside Time	

19

If there is a significant percentage jump, especially if it is sustainable, Investigate root cause. It's these metrics that cause me to go back and compare to (or look for a pattern in) historically captured performance data.

Danger Will Robinson! Danger!

	AVAILABLE	HADR DOWN	HADR CONGESTED	TABLESPACE STATUS	LOG UTILIZATION	CRITICAL FILE SYSTEM	CPU UTILIZATION	SWAP UTILIZATION	DEADLOCK INCREASE
PRD	✓	✓	✓	✓	✓	✓	✓	✓	✓
STG	✓			✓	✓	✓	✓	✓	✓
LT	✓			✓	✓	✓	✓	✓	✓
QA	✓	✓	✓	✓	✓	✓	✓	✓	✓
DEV	✓			✓	✓	✓	✓	✓	✓

20

A dashboard can be tied into a tool backend like DBI's Brother Panther, to data collection from homegrown scripts, or from the Dashboard tool itself. However, be careful of footprint on the database. Repetitive SQL in an inefficient manner against old and deprecated tables can be dangerous.

Historical Efficiency/Performance Metrics

“DB2 Workload Manager as a Monitoring Solution” by Abhik Roy



This is part one of a three part series. The full series includes:

- DB2 Workload Manager (WLM) as a Monitoring Solution – Understanding WLM (Part 1 of 3)
- DB2 Workload Manager (WLM) as a Monitoring Solution– How to Set up WLM (Part 2 of 3)
- DB2 Workload Manager (WLM) as a Monitoring Solution– Analyzing WLM Information (Part 3 of 3)

Normally:

- A feature with Advanced Enterprise Server and Advanced Workgroup Server Editions.
- Normally priorities execution and use of resources.

Alternatively:

- Historically capture performance data such as ..
I/O, Memory, Bufferpools, SortHeap, etc

“DB2 Workload Manager as a Monitoring Solution” by Abhik Roy:

<https://datageek.blog/2015/02/10/db2-workload-manager-wlm-as-a-monitoring-solution-understanding-wlm-part1-of-3/>

<https://datageek.blog/2015/03/17/db2-workload-manager-wlm-as-a-monitoring-solution-how-to-set-up-wlm-part-2-of-3/>

<https://datageek.blog/2015/04/07/db2-workload-manager-wlm-as-a-monitoring-solution-analyzing-wlm-information-part-3-of-3/>

Historical Efficiency/Performance Metrics

“Emulating data reset with the new DB2 9.7 monitoring table functions” by Scott Walkty



Content series:

— This content is part 1 of 2 in the series: Monitoring in DB2 9.7

Part 1: Emulating data reset with the new DB2 9.7 monitoring table functions

Part 2: Relational access to XML event monitor data in DB2 9.7

- Originally written to take advantage of new MON_{TABLE} features in 9.7
- Captures all sorts of performance metrics
- Allows you to compare a historical point in time

“Emulating data reset with the new DB2 9.7 monitoring table functions” by Scott Walkty:

<http://www.ibm.com/developerworks/data/library/techarticle/dm-1009db2monitoring1/>

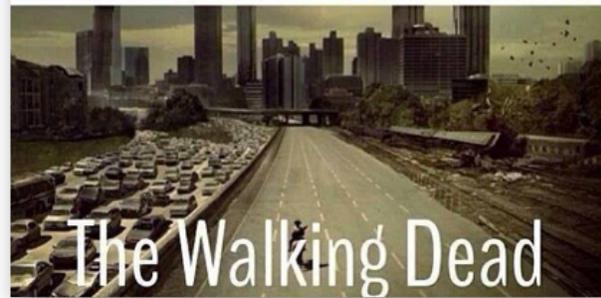
DBI pureSuite



- Small Footprint
- Index Read Efficiency
- Average Result Set Size
- Percentage of Resource Utilization
- 5 Clicks to Solution
- Index Impact Analysis

Bottlenecks

When resource utilization looks like I-85.



When One Problem is Masked by Another Problem

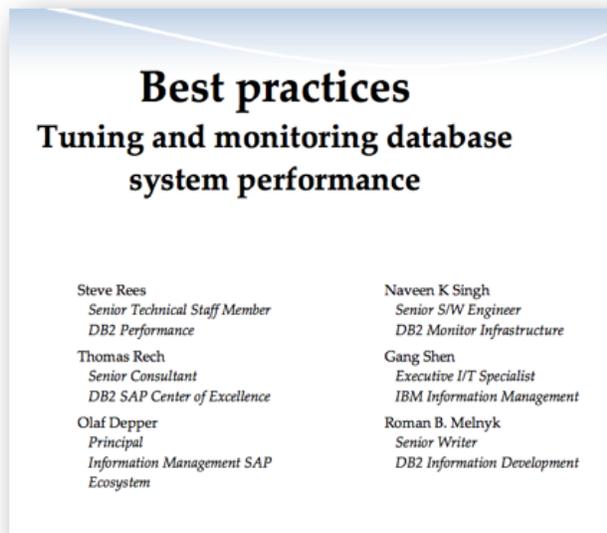
When symptom is not a root cause.

- Network latency shows up as a locking Issue
- Slow internal transaction time shows up as increase in number of connections
- Memory utilization shows up as CPU spike
- Disk contention shows up as a drop in Transactions Per Second

Alerting is reserved for the pain, where our historical data capture shows how we got there.

It helps you correlate the butterfly effect. We stepped on one butterfly (network latency) which caused a locking issue that would not otherwise be there.

Source Used with Permission



The following slides on bottlenecks are heavily sourced, with permission, from best practices documentation written by Steve Rees.

Can be downloaded via developerWorks at this link: <http://tinyurl.com/ldxlarw>

Check Your Bottleneck and Choke Points

"You know, the usual suspects."



Four Types of Bottlenecks

Disk

- High I/O Wait Time (via iostat or vmstat).
- 30% I/O Wait Time could be a clue. If CPU is low as well, could be a strong lead.
- Disk Utilization is over 80%
- Lower to middle CPU Utilization (20-50%)
- High R/W times reported by DB2

Quick Evaluation of Server Level Metrics

Command: IOSTAT [INTERVAL] [COUNT]

```
db2inst1@db2home/db2inst1 $ iostat 2 5
Linux 2.6.32-642.11.1.el6.x86_64 (mhs101.mhsl.local) 03/18/2017 _x86_64_ (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           2.11    0.12   0.74    0.51    0.02   96.50

Device:            tps    blk_read/s    blk_wrtn/s    blk_read    blk_wrtn
xvda                4.52         2.45         55.50    13951168    316510386
xvdb                 0.00         0.02         0.07       93814       383816
dm-0                 5.05        13.34        64.90    76079428    370148248
```

avg-cpu: **%IOWAIT** Percentage CPU was IDLE while waiting I/O to complete
 %IDLE Percentage CPU was IDLE without an outstanding disk request

Device: **TPS** Transfers Per Second (an I/O request to the device)

Four Types of Bottlenecks

CPU

- Overall CPU Saturation
 - 95% is considered saturated
- Single CPU Saturation
 - Symptomatic a specific application or statement is root cause
- System Utilization : User Utilization ratio is unusual
 - [3 (System) : 1 (User)] or [4 (System) : 1 (User)] is not uncommon

Quick Evaluation of Server Level Metrics

Command: `VMSTAT [INTERVAL] [COUNT]`

```
procs -----memory----- ---swap-- -----io----- --system-- -----cpu-----
r b swpd free buff cache si so bi bo in cs us sy id wa st
0 0 7756 357620 382764 30648576 0 0 96 39 1 1 2 1 97 1 0
0 0 7756 357604 382764 30648576 0 0 0 32 455 700 0 0 100 0 0
0 0 7756 357604 382764 30648588 0 0 0 0 445 712 0 0 100 0 0
0 0 7756 357108 382772 30648588 0 0 0 6 451 695 0 0 100 0 0
0 0 7756 357232 382772 30648588 0 0 0 70 437 709 0 0 100 0 0
```

Procs: **R** Processes waiting on Processor; **B** processes in sleep state

Memory: **SWPD** How much memory (in K) swapped out or written to disk
FREE Amount of free memory; **BUFF** Amount of memory in use

IO: **BI** Bits in; **BO** Bits out

CPU: **ID** CPU idle percentage; **WA** CPU wait percentage

This is a snapshot in time based on your wait interval.

CPU Saturation by Workload with Brother-Panther

Brother-Panther®

File Edit View Tools Reports Window Help

Statement Performance for [Statement ID]

Statement Workload from 6/20/18 8:00 AM to 6/20/18 3:00 PM

Last Refresh: 7/6/18 2:10 PM
Rows: 624 Max Rows: 5000

Follow Up	Stmt ID	Verb	Type	# Execs	% CPU Time	% Exec Time	Avg Exec Time (sec)	% Data L Reads	Avg L Reads	% Temp Data L Reads	Avg Temp L Reads	% Rows Read	Avg Rows Read	IX Read Efficiency	% Rows Fetched	Avg Rows Fetched
	99174801...	UPDATE	DYNAMIC	724	44.104%	28.665%	0.371287	48.178%	270,558.624	0.000%	0.000	48.567%	261,470.837	261,470.837	0.000%	0.000
	D852A462...	SELECT	DYNAMIC	724	33.656%	21.905%	0.281019	50.180%	274,734.662	3.176%	3.028	50.650%	270,077.895	1,473.660	12.194%	183.270
	D488C18D...	DELETE	DYNAMIC	1	0.968%	0.510%	0.004630	0.414%	5,083.572	0.000%	0.000	0.372%	2,008.128	2,008.128	0.000%	0.000
	D011D090...	SELECT	DYNAMIC	1	0.000%	0.000%	0.002450	0.000%	237.000	0.000%	0.000	0.000%	89.000	1.000	0.000%	89.000
	FFF1A020...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FF6604E9...	SELECT	DYNAMIC	1,744	0.024%	0.011%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FF80C21C...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FE055179...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	FB054497...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	F8A51468...	SELECT	DYNAMIC	1	0.000%	0.001%	0.000000	0.000%	62.000	0.000%	0.000	1.000%	62.000	1.000	0.006%	62.000
	F9A5177D...	SELECT	DYNAMIC	1	0.000%	0.000%	0.000000	0.000%	0.000	0.000%	0.000	0.000%	0.000	0.000	0.000%	0.000
	F8699302...	SELECT	DYNAMIC	6,539	0.126%	0.056%	0.000000	0.000%	3.851	0.000%	0.000	1.000%	2.286%	1.000	2.286%	3.851



Four Types of Bottlenecks

Memory

- Usually an alternate resource shows the pain because of extra work due to memory shortage
 - Small Log Buffer Size leading to (Log) Disk Bottleneck
 - Small Buffer Pool leading to Data/Index Tablespace Bottleneck
- Swap Space, usually with lower CPU, shows overall memory pressure

Quick Evaluation of Server Level Metrics

Command: top

```
top - 10:54:01 up 65 days, 22:03, 1 user, load average: 0.00, 0.04, 0.00
Tasks: 295 total, 1 running, 292 sleeping, 0 stopped, 2 zombie
Cpu(s): 0.1%us, 0.2%sy, 0.0%ni, 99.7%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 32749888k total, 32396996k used, 352892k free, 384232k buffers
Swap: 2096444k total, 7756k used, 2088688k free, 30650464k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2106	root	20	0	412m	15m	1272	S	0.3	0.0	67:07.68	dthostagent
14071	db2inst2	20	0	6076m	1.8g	1.7g	S	0.3	5.6	141:04.77	db2sysc
24345	root	20	0	17572	2256	1736	S	0.3	0.0	0:07.63	processes
24578	db2inst1	20	0	29736	1584	1084	R	0.3	0.0	0:00.14	top
1	root	20	0	33680	1544	1164	S	0.0	0.0	0:42.81	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.43	kthreadd

Does not show disk level metrics, but does show **top processes**

Easier to quickly interpret (in my opinion)

This is real time and continuously update.

Lazy Bottlenecks

"I'm givin' her all she's got Captain!"

When the system just can't be pushed any further, but the typical bottle neck isn't apparent.

- Lock contention Wait, Escalation, Timeout, Deadlock
- Prefetching isn't sufficient "Data isn't waiting for me, let me go out and get it .."
- Poor page cleaning "Let me stop and write this down before reading more data.."
- Application Rate of requests decreases unexpectedly

Eliminating the Last Suspect (Lazy Bottleneck) - SQL

```
-- SQL (Lock Wait and Lock Chain details):
SELECT
  LOCK_NAME,
  HLD_APPLICATION_HANDLE as AGENT_BLOCKING_LOCK, REQ_APPLICATION_HANDLE as AGENT_WANTING_LOCK,
  LOCK_STATUS as CURRENT_STATUS, LOCK_WAIT_START_TIME as LOCK_WAIT_START
FROM TABLE (MON_GET_APPL_LOCKWAIT(NULL, -2));
```

* SQL in Student Notes

LOCK_NAME	AGENT_BLOCKING_LOCK	AGENT_WANTING_LOCK	CURRENT_STATUS	LOCK_WAIT_START
0200184E00000000000000000000000054	5317	7245 W		2017-04-17-12.29.24.001203

```
-- SQL (Object in contention):
SELECT
  SUBSTR(NAME,1,20) AS OBJ_DETAIL,
  SUBSTR(VALUE,1,50) AS VALUE
FROM TABLE ( MON_FORMAT_LOCK_NAME('<Lock Name>')) as LOCK;
```

OBJ_DETAIL	VALUE
LOCK_OBJECT_TYPE	TABLE
TBSP_NAME	USERSPACE1
TABSHEMA	DB2INST1
TABNAME	MIKE

Multiple LOCK Queries in Triage repository of my GitHub Account:
 MKrafick/Triage - <https://github.com/MKrafick/Triage>

Eliminating the Last Suspect (Lazy Bottleneck)

Command: `db2top -d DBNAME` (Then U: Locks / L: Lock Chain)

```

C:\DB2\BIN>db2top -d DBNAME,refresh=1secs(0.001)
[db2top]
Locks held.....:      5 [0.00]
Agents waiting...:      0
Applics Connected.:    13

Agent  Application      Application      Object      Lock      Object      Lock      Lock
Id(State) Name                Status          Name        Mode      Type        Status     Count
-----
12(-) db2fw0            Connected       DBA.ROS_LOCKS IX          Table      Granted    1
13(-) db2fw1            Connected       DBA.ROS_LOCKS IX          Table      Granted    1
14(-) db2fw2            Connected       DBA.ROS_LOCKS IX          Table      Granted    1
15(-) db2fw3            Connected       DBA.ROS_LOCKS IX          Table      Granted    1
8793(L) db2jcc_application UOW Waiting in the application Internal Plan S          Plan       Granted    1

-]20:31:03,refresh=1secs(0.001)
[d=Y,a=N,e=N,p=ALL]

Blocker->Blocked Agent Chain
-----
38->40
38->46

Quit: q, Help: h                               Lock=5 (Entries=5), L: Lock Chain
db2top 2.3

```

Screenshot used with permission –datageek.blog (Ember Crooks)

DB2 Bottleneck

Command: `db2top -d DBNAME (Then B: Bottleneck)`

Server Resource	Top Agent	Resource Usage	Resource Value	Application Name
→ Cpu	N/A	0%	0	N/A
→ SessionCpu	108	9.60%	455.909424	db2pcsd
→ IO r/w	17019	40.06%	7,836	db2jcc_application
→ Memory	12727	2.00%	2.2M	db2jcc_application
→ Locks	17020	71.43%	5	db2jcc_application
→ Sorts	14878	45.23%	218	db2jcc_application
→ Sort Time	15331	7.19%	3,132	db2jcc_application
→ Log Used	16238	67.95%	10.7K	db2jcc_application
→ OverFlows	10499	8.02%	30	db2jcc_application
→ RowsRead	15633	45.76%	6,935	db2jcc_application
→ RowsWritten	17019	97.73%	1,118	db2jcc_application
→ TQ r/w	N/A	0%	0	N/A
→ MaxQueryCost	17019	99.91%	122,375	db2jcc_application
→ XDAPages	N/A	0%	0	N/A

Atypical Lazy Bottleneck (Mike's Observation)

Consider TABLESPACE STATE as a Lazy or Disk Bottleneck.

- Application (or Users) can suddenly do one task but not another
- Usually locks out access

BACKUP PENDING

After load, newly roll forward recovery

QUIESCED

Exclusively, prevents anyone from accessing

OFFLINE or NOT ACCESSIBLE

Disk level or container issue

Symptom:

Intermittent access for one query but not another, one user but not another, application receives SQL error code

Catching Tablespace State Issue

Query on a schedule, or a “as needed” script in triage tool box.

Build a dynamic script around three lines:

```
db2 -x "select substr(tbsp_name,1,20), substr(TBSP_STATE,1,20) from table
(MON_GET_TABLESPACE('','-2'))" > $SCRIPTPATH/tbsp.tmp

TEST_TS=`cat $SCRIPTPATH/tbsp.tmp | grep -v
'NORMAL\|BACKUP_IN_PROGRESS\|LOAD_IN_PROGRESS\|REORG_IN_PROGRESS' | wc -l`

if [ $TEST_TS -gt 0 ]; then
    WHATEVER ALERTING OUTPUT YOU NEED
fi
```

* SQL in Student Notes

41

Tablespace State Hook available at the DBMonitoring repository of my
 GitHub Account:

DiscoverDB2/BAD_TABLESPACE_HOOK.sql -
https://github.com/MKrafick/DBMonitoring/blob/master/BAD_TABLESPACE_HOOK.sql

Hexadecimal Values and various states can be found here – Knowledge
 Center: Tablespaces States

https://www.ibm.com/support/knowledgecenter/SSEPGG_10.5.0/com.ibm.db2.luw.admin.dbobj.doc/doc/c0060111.html

Triage The Problem

The database administrators crash cart.

First Tool – Question Everything

“Don’t look for a needle in a needlestack.”

Who picked up the problem?

Alerting, end user, routine audit, etc

What is the symptom?

App. slow down, end user blank page, etc

When did this happen?

Timebox the problem

Where did this happen?

Database, server, environment

Do you have details?

Give me an application error message, etc

DB2 Error Log – Listen to What The Patient is Telling You

```
db2 get dbm cfg | grep DIAG
Diagnostic error capture level          (DIAGLEVEL) = 3
Diagnostic data directory path         (DIAGPATH)  = /db2diag/
```

Old Standby:

```
cat db2diag.log | tail -100 db2diag.log | vi db2diag.log
```

Make it easy on yourself:

```
db2diag -g level=Error (Info, Warning, Error, Severe, Critical, Event)
```

Showing Off: * SQL in Student Notes

```
SELECT TIMESTAMP, LEVEL, IMPACT, substr(DBNAME,1,10) as DBNAME,
substr(APPL_ID,1,25) as APPL_ID, substr(AUTH_ID,1,20) as AUTH_ID, substr(MSG,1,50)
as MESSEAGEROM TABLE (PD_GET_DIAG_HIST( 'MAIN', 'ALL', '', NULL, NULL ) ) AS T"
```

44

Diag Parsing Script available at the DISCOVER.DB2 repository of my GitHub Account:

DiscoverDB2/PARSE_DIAG.ksh -
https://github.com/MKrafiCK/DiscoverDB2/blob/master/PARSE_DIAG.ksh

Discover.DB2 Video (16:05) "Exploring the Db2 Error Log" -
<https://youtu.be/EOZrJUludRE>

Note:

Db2diag.log is normally found `$INSTHOME/sql/lib/db2dump`

DIAGLEVEL:

- 0 – No diagnostic data captured
- 1 – Severe errors only
- 2 – All errors
- 3 – All errors and warnings
- 4 – All errors, warnings and informational messages

Information in the DB2DIAG.log

```

2017-02-17-03.21.26.068716-360 I4761E590 LEVEL: Error
PID : 11640 TID : 140265222301440 PROC : db2sysc 0
INSTANCE: db2inst1 NODE : 000 DB : DBNAME
APPHDL : 0-19101 APPID: 00.000.00.0]59488.170217092058
AUTHID : OMSUSR HOSTNAME: devbox.mhsl01.mhsl.local
EDUID : 729 EDUNAME: db2agent (DBNAME) 0
FUNCTION: DB2 UDB, buffer pool services, sqlbCheckStateTransition, probe:0
MESSAGE : ZRC=0x80020038=-2147352520=SQLB_INVALID_STATE_TRANSITION
"Invalid state transition"
  
```

Level : Info, Warning, Error, Severe, Critical, Event

Proc(ess): Process name

APPID: Will match "List Applications", can show IP of incoming application with problem

AUTHID: ID running into the problem

Message: Verbose feedback on error.

Can contain a SQL Error Code, SQLSTATE, RC (Return Code), and ZRC Codes

Interpreting the DB2DIAG.log

MESSAGE: ZRC=0x80020038=-2147352520=SQLB_INVALID_STATE_TRANSITION
 "Invalid state transition"

Command: db2diag -rc {ZRC Code}

Display description of DB2 internal
 Return Codes of ZRC hexadecimal return
 codes.

```
db2diag -rc 0x80020038
Input ZRC string '0x80020038' parsed as 0x80020038 (-2147352520).
ZRC value to map: 0x80020038 (-2147352520)
V7 Equivalent ZRC value: 0xFFFF8138 (-32456)

ZRC class :
  SQL Error, User Error,... (Class Index: 0)
Component:
  SQLB ; buffer pool services (Component Index: 2)
Reason Code:
  56 (0x0038)

Identifier:
  SQLB_INVALID_STATE_TRANSITION
Identifier (without component):
  SQLB_RC_BADTRAN

Description:
  Invalid state transition

Associated information:
  [Sqlcode -291]
  SQL0291N State transition not allowed on table space.

Number of sqlca tokens : 0
Dialog message number: 1
```

Interpreting the DB2DIAG.log

SQL0291N State transition not allowed on table space.

But, but ... I need more.

Getting a detailed message:

- Knowledge Center
- Command Line Lookup

db2 ? SQL0291N



```
db2 ? SQL0291N
```

```
SQL0291N State transition not allowed on table space.
```

```
Explanation:
```

```
An attempt was made to change the state of a table space. Either the new state is not compatible with the current state of the table space, or an attempt was made to turn off a particular state and the table space was not in that state.
```

```
User response:
```

```
Table space states change when a backup is taken, the load completes, the rollforward completes, etc., depending on the current state of the table spaces. Refer to the systems administration guide for further information about the table space states.
```

```
sqlcode: -291
```

```
sqlstate: 55039
```

Notice the comment:

“Table Space states change when a backup is taken, the load completes”

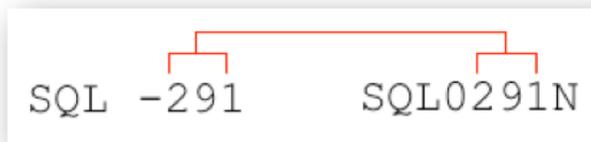
Interpreting the DB2DIAG.log

Associated information: Sqlcode -291

SQL0291N State transition not allowed on table space.

Quick Tip:

When you don't have a SQL Error Code (like SQL0291N) that you can look up, extrapolate the code from a negative SQL Code.



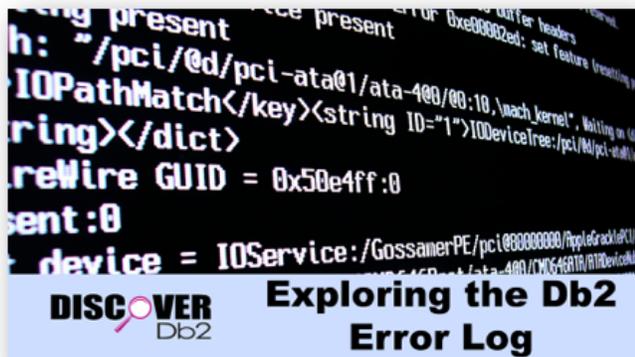
In the end, what happened?

```

2017-02-17-03.21.26.211568-360 I6313E630 LEVEL: Severe
PID : 11640 TID : 140265222301440 PROC : db2sysc 0
INSTANCE: db2inst1 NODE : 000 DB : DBNAME
APPHDL : 0-19101 APPID: 00.000.00.0.59488.170217092058
AUTHID : OMSUSR HOSTNAME: devbox.mhsl01.mhsl.local
EDUID : 729 EDUNAME: db2agent (DBNAME) 0
FUNCTION: DB2 UDB, buffer pool services, sqlbStateGetCheck, probe:633
DATA #1 : String, 22 bytes
Current pool state is:
DATA #2 : Pool State, PD_TYPE_SQLB_POOL_STATE, 4 bytes
0x00000800
- SQLB_BACKUP_IN_PROGRESS
  
```

In this specific case, two more error messages proceeded after the “Invalid State Transition”

1. Backup is in Progress
2. An Error with a REORG Statement



www.youtube.com/DiscoverDB2



DISCOVER.DB2

Published on May 1, 2018

The error log can be intimidating. Sometimes the error log seems to have too much information to parse, sometimes it doesn't seem to have enough. This episode will cover how to find the error log, how to effectively mine information found inside, and some oddball tips to help you along.

Timestamps:

- 00:44 - Where is the error log?
- 02:16 - What is in the error log?
- 03:43 - Cool ways to look up error codes
- 06:35 - Command line walkthrough
- 09:59 - How to use PD_GET_DIAG_HIST

GitHub Repository:

<https://github.com/MKrafcik/DiscoverDB2>

PARSE_DIAG.ksh

A quick script that takes a database name and pulls 24 hours of DB2DIAG.log details via PD_GET_DIAG_HIST.

Useful Links:

- (Blog) DB2 Error Logging - <https://tinyurl.com/ybxgz8>
- (Blog) db2diag tool for Parsing the Diagnostic Log - <https://tinyurl.com/7q3pnqu>

Category [Science & Technology](#)

YouTube Channel: <https://www.youtube.com/DiscoverDB2>

Quick Evaluation of DB2 Metrics - MONREPORT.DBSUMMARY

Give me ALL THE METRICS!

MONREPORT.DBSUMMARY

Syntax:

```
db2 "call monreport.dbsummary (No. Seconds)"
```

What it does:

Calls table functions, waits X seconds, calls table functions again. Calculate the difference.

Summarizes:

System Performance | Application Performance | Member Info

Quick Evaluation of DB2 Metrics - MONREPORT.DBSUMMARY

Buffer pool hit ratios

Type	Ratio	Formula
Data	99	$(1 - (20 + 0 - 0)) / (155402 + 860)$
Index	99	$(1 - (33 + 0 - 0)) / (205472 + 3966)$

- I/O wait time is
(POOL_READ_TIME + POOL_WRITE_TIME + DIRECT_READ_TIME + DIRECT_WRITE_TIME).

MEMBER	TOTAL_CPU_TIME per request	TOTAL_ WAIT_TIME %	RQSTS_COMPLETED TOTAL	I/O wait time
0	135	1	33191	83

Quick Evaluation of DB2 Metrics - MONREPORT.DBSUMMARY

Locking

	Per activity	Total
LOCK_WAIT_TIME	0	0
LOCK_WAITS	0	0
LOCK_TIMEOUTS	0	0
DEADLOCKS	0	0
LOCK_ESCALS	0	0

Row processing

ROWS_READ/ROWS_RETURNED = 10 (880348/86471)
ROWS_MODIFIED = 27434

VM, Cloud Computing, Noisy Neighbors and Server Level Metrics

One application uses the majority of resources negatively affecting a second application.

- Virtual Servers, or LPAR's (Logical Partitions), where you have shared resources.
- Can complicate or mask root cause analysis

Although any of the main resources can be affected, in my experience **it makes disk metrics suspect.**

- I/O Per Second (IOPS)
- Disk Queue Length
- Reads/Writes; Random/Sequential
- I/O Section of MONREPORT.DBSUMMARY

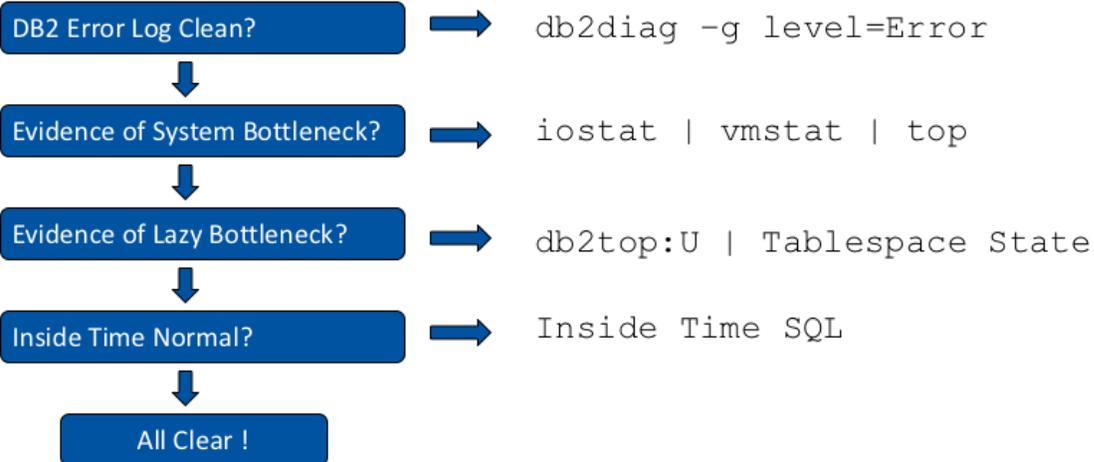


Be wary when storage team touts load balancing, hotspot management, cache, etc.

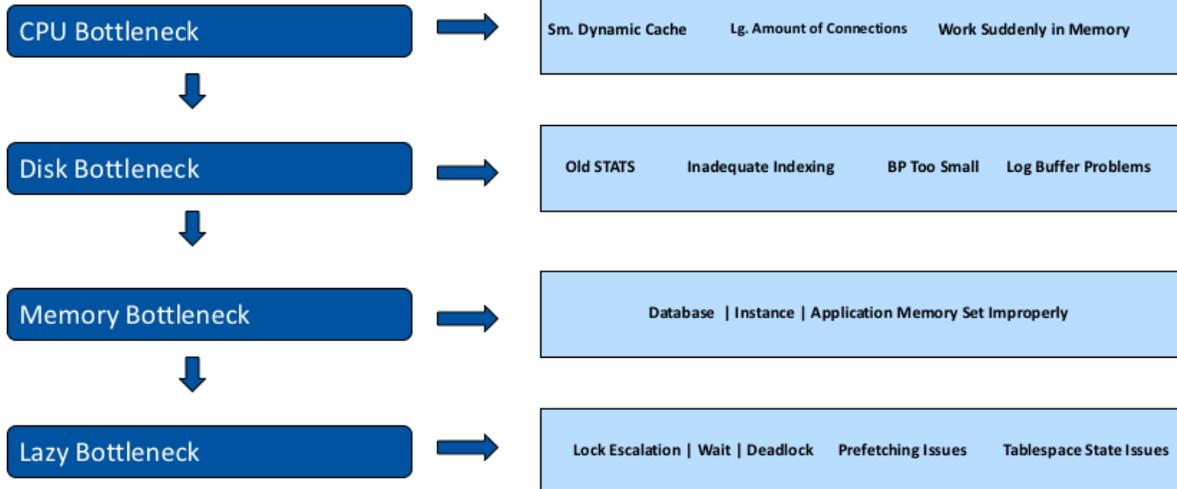
Assessing the Problem

"Well, there's your problem!" – Adam Savage

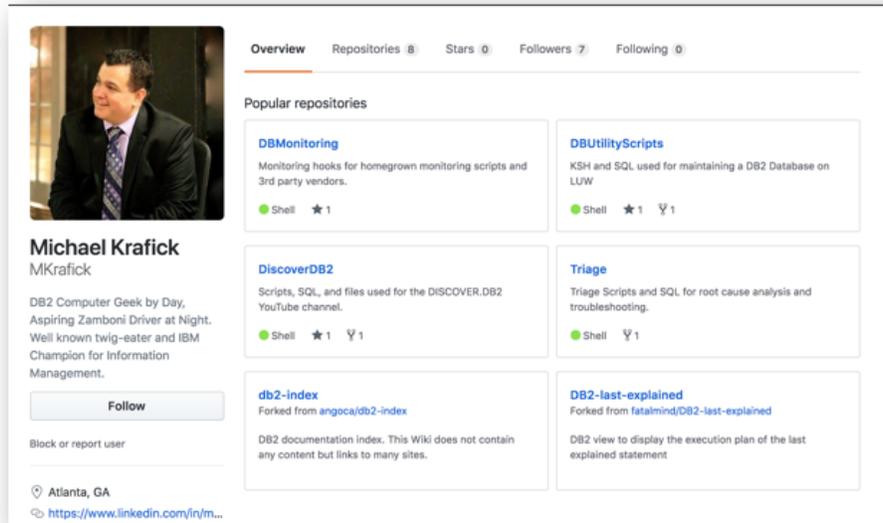
“All Clear” or “Captain, we have a problem”?



"We Have a Problem"



Resources



Michael Krafcik
MKrafcik

DB2 Computer Geek by Day,
Aspiring Zamboni Driver at Night.
Well known twig-eater and IBM
Champion for Information
Management.

[Follow](#)

Block or report user

Atlanta, GA
<https://www.linkedin.com/in/m...>

Overview Repositories 8 Stars 0 Followers 7 Following 0

Popular repositories

- DBMonitoring**
Monitoring hooks for homegrown monitoring scripts and 3rd party vendors.
Shell ★ 1
- DBUtilityScripts**
KSH and SQL used for maintaining a DB2 Database on LUW
Shell ★ 1 1
- DiscoverDB2**
Scripts, SQL, and files used for the DISCOVER.DB2 YouTube channel.
Shell ★ 1 1
- Triage**
Triage Scripts and SQL for root cause analysis and troubleshooting.
Shell 1
- db2-index**
Forked from [angeca/db2-index](#)
DB2 documentation index. This Wiki does not contain any content but links to many sites.
- DB2-last-explained**
Forked from [fataimind/DB2-last-explained](#)
DB2 view to display the execution plan of the last explained statement

Krafcik's Github - <https://github.com/Mkrafcik>



.gitattributes	.gitattributes	ALL_LOCKING_EVENTS.sql
CREATE_EXPLAIN.ksh	CREATE_EXPLAIN.ksh	CARD_ACTUALS.ksh
CRONTAB_TEMPLATE	CRONTAB_TEMPLATE	LOCK_COUNT_BY_HOUR.sql
DB2SET.ksh	DB2SET.ksh	LOCK_WAIT_CHAIN.sql
DISABLE_HEALTH_MONITOR.ksh	DISABLE_HEALTH_MONITOR.ksh	README.md
EXT_STATUS.sql	EXT_STATUS.sql	STMTS_IN_DEADLOCKS.sql
IREF.sql	IREF.sql	SUMMARIZE_BY_STATEMENT.sql
PARSE_DIAG.ksh	PARSE_DIAG.ksh	SUM_LOCK_EVENTS.sql
PROFILE_TEMPLATE	PROFILE_TEMPLATE	SUM_TABLE_LOCK_EVENT.sql
README.md	README.md	TOP20SQL.ksh
REVOKE_PUBLIC.ksh	REVOKE_PUBLIC.ksh	
UPDATE_DBM_PREF.ksh	UPDATE_DBM_PREF.ksh	
UPDATE_DB_PREF.ksh	UPDATE_DB_PREF.ksh	

Krafick's Github - <https://github.com/Mkrafick>

Resources

DISCOVER
Db2

DISCOVER.DB2
181 subscribers

CUSTOMIZE CHANNEL CREATOR STUDIO

HOME **VIDEOS** PLAYLISTS CHANNELS ABOUT

Uploads PLAY ALL SORT BY

- DISCOVER.DB2** Dockerfile for Db2 (Part I) 14:19
(Ep. 25) - Writing a Dockerfile for Db2 (Part I)
2 views • 7 hours ago
- DISCOVER.DB2** Connectivity Export Commands 8:23
(Ep. 23) - Exporting Connectivity Settings
21 views • 3 weeks ago
- DISCOVER.DB2** DB2LOG 3:35
(Ep. 22) - DB2LOOK
43 views • 4 weeks ago
- DISCOVER.DB2** Db2 High Watermark and Space Red 16:34
(Ep. 21) - Db2 REDUCE MAX Command
78 views • 1 month ago
- DISCOVER.DB2** Db2 REORG 13:32
(Ep. 20) - Db2 REORG
133 views • 1 month ago



10 Minute Triage: Troubleshooting Production Issues 101

Michael Krafiick (Twitter: @mkrafiick)
Lead Db2 Database Engineer, Atlanta

Session Code: H6
Thursday, September 13, 2018 (9:45 am) | Platform: LUW

