Entspannte Releases mit DB-Workloadtests

PASS DE RG Stuttgart

07/23

Download unter

https://tinyurl.com/passrm05

ÜBER MICH

- Martin Guth (40)
- 14 Jahre BI-Entwickler (Aufbau eines neuen DWH)
- 7 Jahre DBA (Schwerpunkt Performancetuning)
- Aktuelle Rolle „Data Swiss-Knife“
  bei 3C Deutschland GmbH (Experian) in Heilbronn
- Begeistert von der PASS seit 2008

- Blog: www.martinguth.de
- Kontakt: martin_guth@hotmail.com

TEAM HUGO

Agenda

1. Worum geht‘s eigentlich?
2. Workloadtesting mit den Workload Tools
3. Demos
4. Tipps & Tricks
5. Lessons Learned
6. Fragen und Diskussion

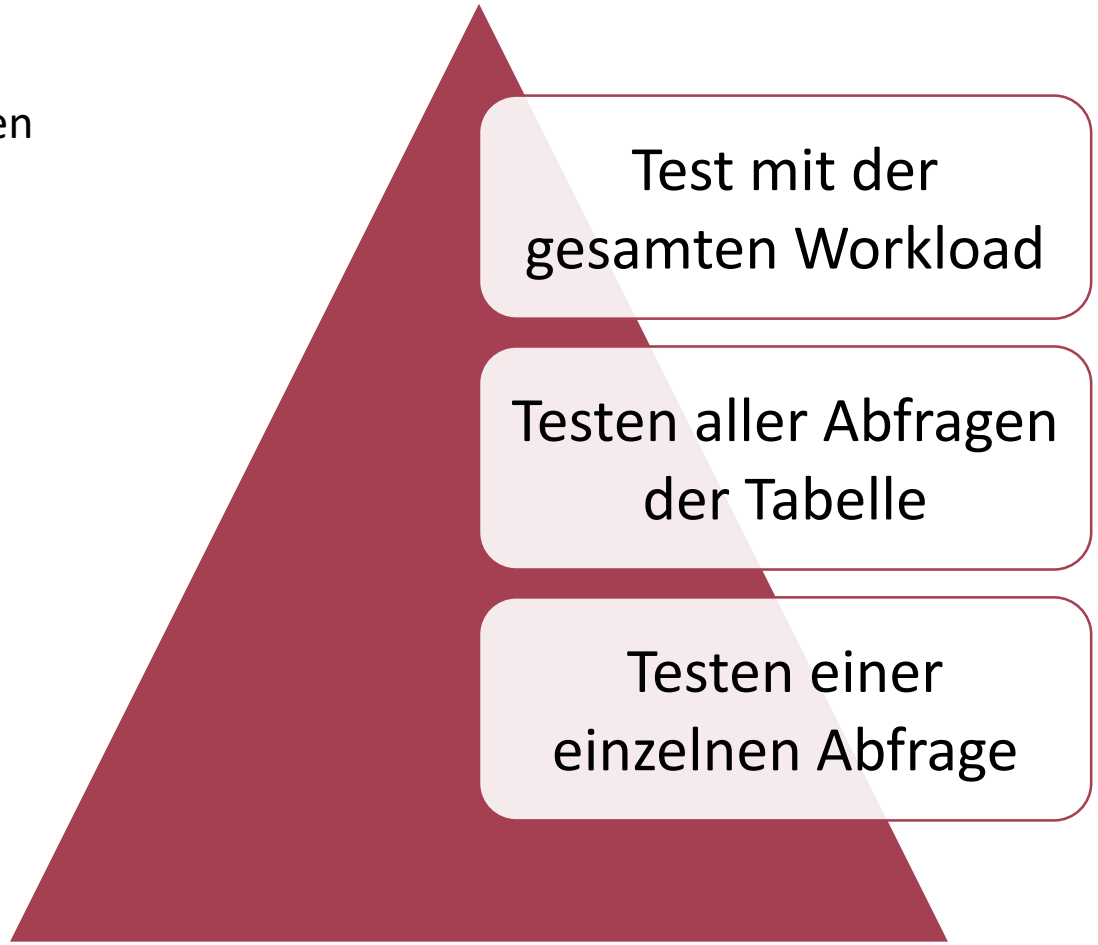
WORUM GEHT‘S EIGENTLICH?

Kleine Änderungen Große Auswirkungen

- Beispiel Performancetuning: Indexanpassung
  - Abfragen / DML-Operationen werden schneller
  - Abfragen / DML-Operationen werden langsamer
- Andere Beispiele
  - Nutzung neuer Features (z.B. ADR)
  - Optimierung von DB-Einstellungen (z.B. MAXDOP)
- Handlungsoptionen
  - No Risk no Fun
  - Never change a running System
  - Vorab umfassend testen

Testen Ja, aber wie?

Workload = alles, was auf einem Server innerhalb einer bestimmten Zeit abgearbeitet wird, das heißt Lese- und Schreiboperationen



Test einer einzelnen Abfrage

- Abfrage verändern und Ausführungsergebnisse vergleichen
- Für Indexanpassungen: Test auf separater Instanz mit vergleichbarer Datenmenge und identischem Schema
- Vorteile
  - Am einfachsten umzusetzen
  - Wenig Zusatzaufwand für Testen
- Nachteile
  - Auswirkungen auf andere Abfragen oder DML-Operationen unbekannt

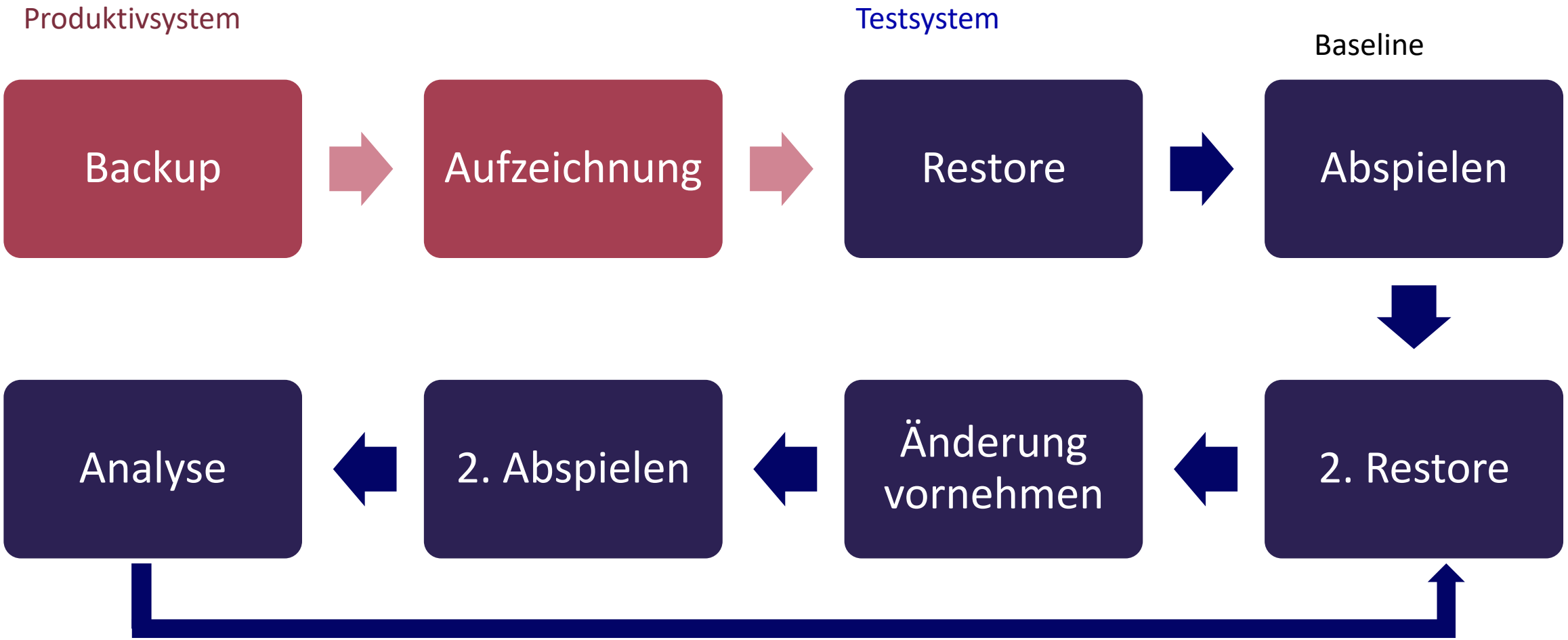
Test aller Abfragen der Tabelle

- Sinnvoll bei Indexanpassungen
- Test auf separater Instanz mit vergleichbarer Datenmenge und identischem Schema
- Gewinnen aller Abfragen zu einer Tabelle aus
  - Plan Cache
  - Query Store (ab SQL Server 2016)
- Vorteile
  - Abschätzen der Auswirkungen auf andere Abfragen möglich
- Nachteile
  - Test von DML-Operationen nicht einfach möglich
  - Oft nicht alle Abfragen protokolliert (Trivial Plans, Query Store Capture Mode)

Test aller Abfragen der Tabelle (2)

- Vorgehensweise
    1. Ausführung ohne Veränderung (Baseline)
    2. Vornehmen von Änderungen
    3. Erneutes Ausführen
    4. Vergleich von ausgewählten Metriken

Ablauf eines DB-Workload-Tests (A/B-Test)



Ein paar Worte zum Testsystem

- Sizing
  - Genügend Plattenplatz (kann langsamer sein)
  - RAM-Größe identisch zu Produktion
  - CPU-Konfiguration identisch zu Produktion
- SQL-Server Instanzoptionen (z.B. MAXDOP) identisch zu Produktion
- Erforderliche Sorgfalt: DSGVO, Security
- Notlösung
  - Separate DB auf Produktion
  - Workloadtest in Randzeiten (z.B. Wochenende)
  - ACHTUNG: Sicherstellen, dass Workloadtest nicht auf Produktion zugreift (z.B. eigener Login ohne Rechte auf Produktion)

WORKLOADTESTING MIT DEN WORKLOAD TOOLS

Workload Tools

- Entwickelt von MVP Gianluca Sartori (spaghettidba)
- Aufzeichnung über Profiler, Trace oder XEvents
- Unterstützt SQL Server ab Version 2000 (getestet ab 2005)
- Speicherformat: SQLite DB
- Abspielen über integriertes Tool
- Download über Github

Workload Tools: Bestandteile

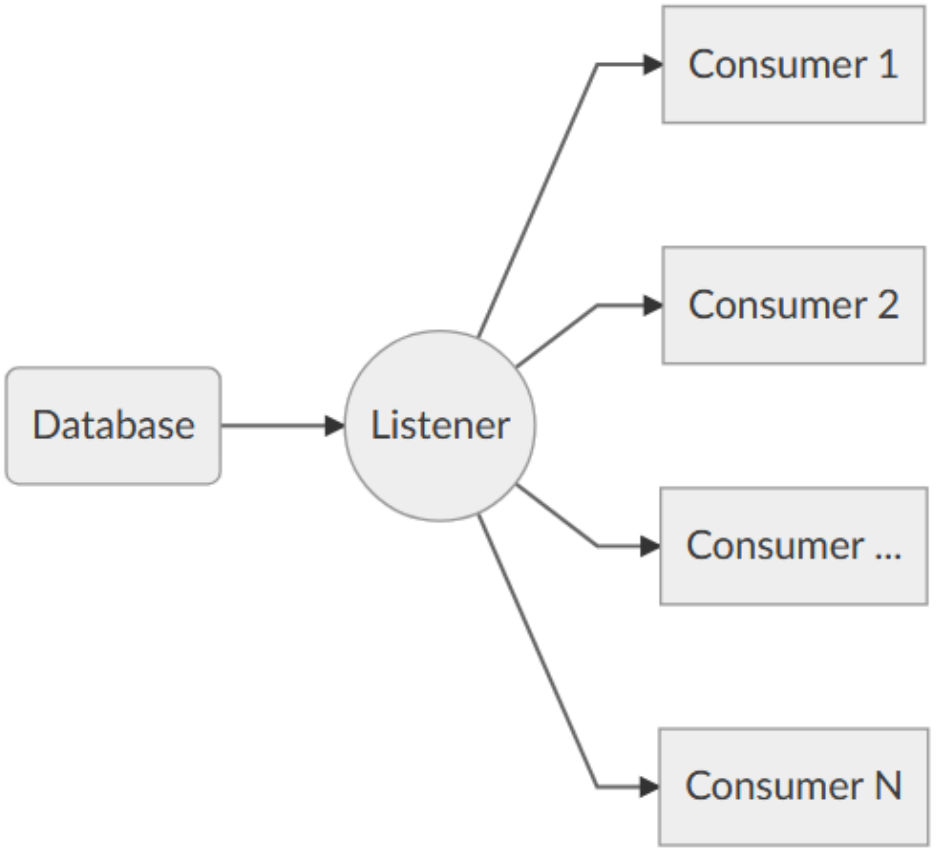
SqlWorkload	WorkloadViewer	ConvertWorkload	WorkloadWizard
• Aufzeichnen und Abspielen von Workloads • Kommandozeile	• Vergleich von Testergebnissen • Grafische Oberfläche	• Konvertierung von Tracefiles • Kommandozeile	• vsl. grafische Oberfläche für einfachere Konfiguration • gibt‘s noch nicht

Workload Tools: Konfiguration

Listener (immer einer)

Aufzeichnung von Quelle
- ExtendedEventsWorkloadListener
- ProfilerWorkloadListener
- SqlTraceWorkloadListener

Lesen einer Aufzeichnung
- FileWorkloadListener



Consumer (0..n)

Schreiben der Aufzeichnung
- WorkloadFileWriterConsumer

Sammeln von Analysedaten
- AnalysisConsumer

Abspielen der Workload
- ReplayConsumer

JSON-Format

Quelle: spaghettidba.com

Workload Tools: Beispielkonfiguration Aufzeichnung

```
{    "Controller": {
       "Listener":
       {
           "__type": "ExtendedEventsWorkloadListener",
           "ConnectionInfo":
           {
               "ServerName": "mydbserver"
           },
           "DatabaseFilter": "mydb"
       },
       "Consumers":
       [
           {
               "__type": "WorkloadFileWriterConsumer",
               "OutputFile": "C:\\temp\\mydb.sqlite"
           }       ]    }}
```

Erläuterung:

① Aufzeichnung der Workload mittels XEvents

② Verbindung zum Server *mydbserver* über Windows-Authentifizierung

③ Aufzeichnung nur für Datenbank *mydb* (optional)

④ Speichern des Workloads in einer SQLite DB für späteres Abspielen

Vollständige Referenz der Konfigurationsoptionen auf Github

Workload Tools: Arbeitsweise Abspielen

- Pro Session (SPID) wird ein eigener ReplayWorker-Prozess gestartet
- Dieser arbeitet die Befehle für diese Session nacheinander ab
- Mögliches Problem: Abarbeitungsreihenfolge beim Abspielen kann auf dem Testsystem abweichen
  - Prozesse laufen unabhängig von einander
  - Beispiel: SPID 4711 INSERT Datensatz; SPID 4811 UPDATE desselben Datensatzes
  - Beim Replay kann FK-Verletzung auftreten, wenn Replay für SPID 4811 schneller ist als für SPID 4711

Workload Tools: Arbeitsweise Abspielen (2)

- Synchron (synchronizationMode = true)
  - Berücksichtigung von Pausen im Workload
  - Abspieldauer entspricht ungefähr der Aufnahmedauer

- Asynchron (synchronizationMode = false)
  - Befehlsausführung so schnell wie möglich
  - Abspieldauer evtl. deutlich kürzer als Aufnahmedauer
  - Höhere Auslastung des Testsystems
  - Höhere Wahrscheinlichkeit für Fehler beim Replay durch stark abweichende Ausführungsreihenfolge (z.B. FK-Verletzungen)

Workload Tools: Beispielkonfiguration Abspielen

```
{    "Controller": {
       "Listener":
       {
           "__type": "FileWorkloadListener",
           "Source": "C:\\temp\\mydb.sqlite",
           "SynchronizationMode": "true"
       },
       "Consumers":
       [
         {
            "__type": "ReplayConsumer",
            "ConnectionInfo":
            {
               "ServerName": "replayServer",
               "DatabaseName": "mydb"
            },
           "ConsumeResults": "false",
           "QueryTimeoutSeconds": 180
         }
```

①

②

③

④

```
{
   "__type": "AnalysisConsumer",
   "ConnectionInfo":
    {
      "ServerName": "replayServer",
      "DatabaseName": "WorkloadAnalysis",
      "SchemaName": "baseline"
    },
    "UploadIntervalSeconds": 60
}]    }}
```

⑤

Erläuterung:

① Auslesen des Workloads aus vorheriger Aufzeichnung

② synchrones Abspielen

③ Abspielen auf *replayServer* in *mydb* (integrierte Auth.)

④ Zusätzliche Konfiguration für Abspielen (optional):

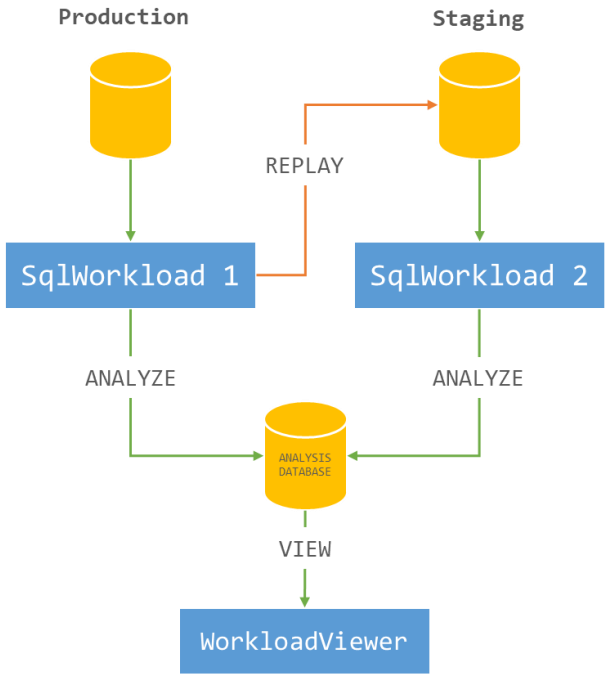
- Verwerfen der Ergebnisse durch den Client (consumeResults *false*)
- Timeout eines Befehls erfolgt nach 3 Minuten

⑤ Analyse des Workloads mit Schreiben von Analyseergebnissen in das Schema *baseline* einer separaten DB alle 60 Sekunden

Vollständige Referenz der Konfigurationsoptionen auf Github

Konfiguration: Entdecke die Möglichkeiten

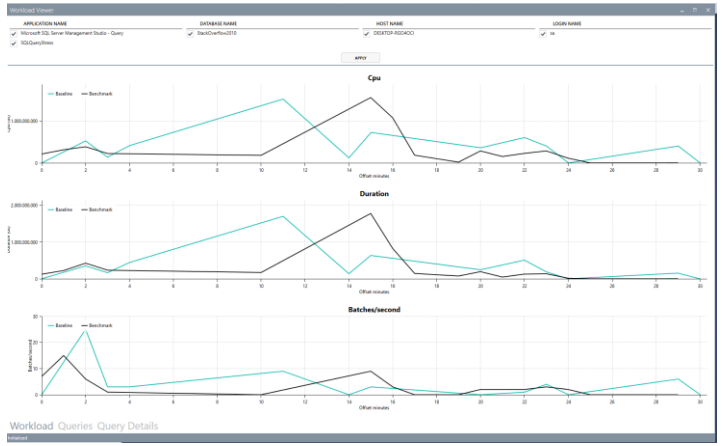
- Es sind noch weitere Konfigurationen möglich
  - Gewinnen von Analyseinformationen direkt von der Produktion → Mutter aller Baselines
  - Real-Time-Replay



Quelle: spaghettidba.com

Analyse der Workloadtests

- WorkloadViewer von Gianluca Sartori
  - In WorkloadTools integriert
  - Kernmetriken CPU-Zeit und Laufzeit
  - (noch) keine Ausführungspläne
- Power BI
  - Vergleichbar zu WOrkloadViewr
  - Anpassbar
  - Mehr Dasbhboardlike
- Analyseskript Query Store von John Sterrett
  - Auch Analyse von Reads/Writes möglich
  - Ausführungspläne aus QueryStore abrufbar

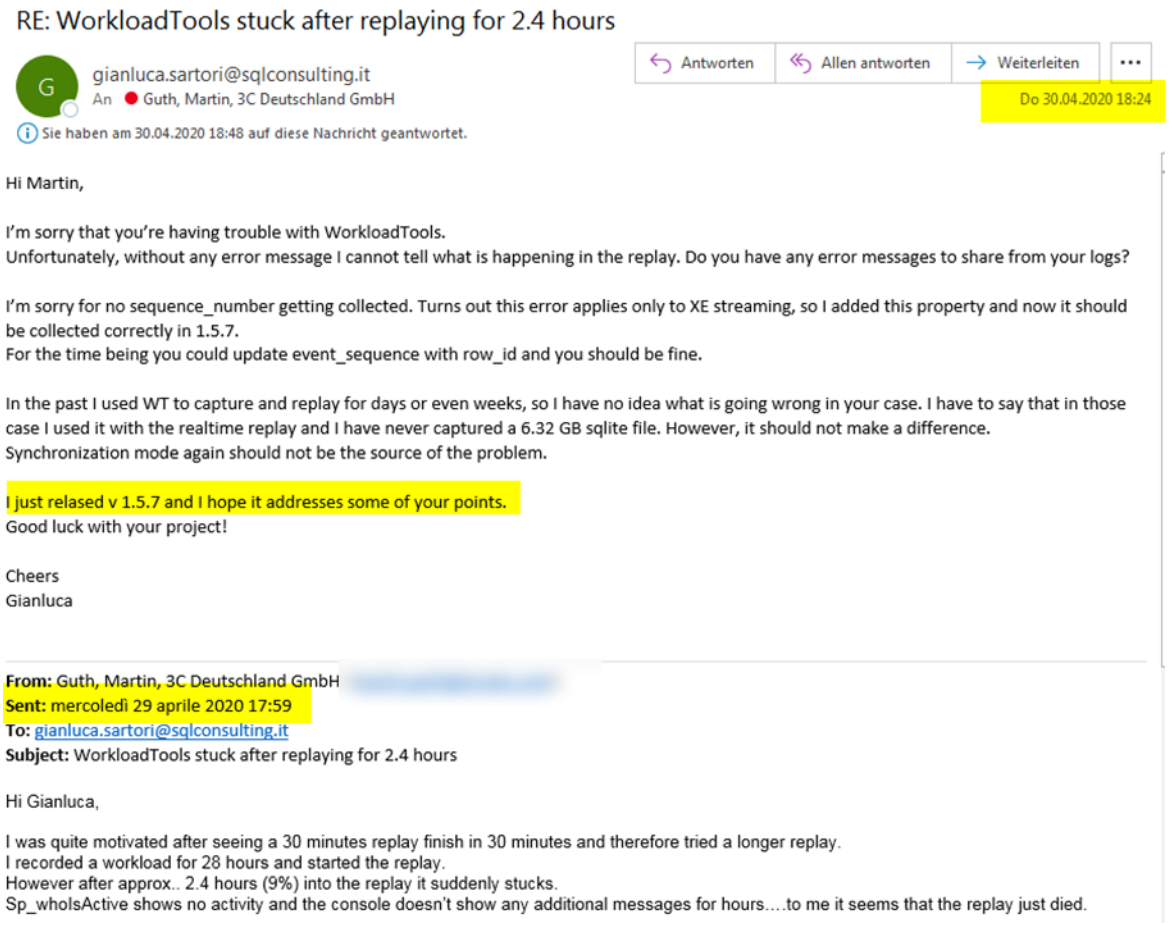


ExecutionD...	CompareExecutions	BaselineExecutions	DurationDelta	CPUDelta	ReadDelta	WriteDelta	CompareReads	BaselineReads	CompareCPU	BaselineCPU	CompareDuration	BaselineDuration	query_hash
0	1767	1767	-12053358	-11903430	-25271941	0	264366	25536307	2953516	14856946	4364968	16418326	0x6438E6BF71C4803B
0	1852	1852	-553623498	-78045527	-4780080	-160	703384	5483464	40556040	118601567	52823103	606446601	0xA113FB5A17EF3178
0	233	233	-251657794	261882578	-3259908	4	8661903	11921811	598405809	336523231	107232174	358889968	0x10B583976D0C9236
0	1032	1032	-18036467	-318656	-1618626	-129	25687755	27306381	17704250	18022906	30964475	49000942	0xA2B99CCEE3D8BDAA
0	10	10	-1056570	-1045266	-1568211	188	60514	1628725	426665	1471931	429464	1486034	0x24161325E430E58F
0	2110	2110	-160298299	225964768	-1470963	0	336332698	337803661	783307080	557342312	462053775	622352074	0xF3806AB866D675B6
0	5738344	5738344	-1786884	-2002862	-1111501	-58536	7308981	8420482	106641230	108644092	107676557	109463441	0x174B21F541E8EE1B
0	77638	77638	918103	-2602495	-1060178	0	1040605	2100783	13626641	16229136	47252596	46334493	0x8D3A11BE85E403EE
0	1027	1027	-660475	-1005781	-827656	1	2500845	3328501	24499825	25505606	25502623	26163098	0x7720AAA63CC6F5C2
0	16	16	-2070010	-12602682	-476589	0	8181248	8657837	11218099	23820781	27000301	29070311	0x7F1ADC76D4F3D9CE
0	2	2	-9885624	-2403985	-394093	0	2289156	2683249	2340884	4744869	2869153	12754777	0xA6DA6495969D7CF3
0	24953	24953	40361021	-5037302	-235511	-6339	35677917	35913428	117990782	123028084	323677505	283316484	0x14E5E89315BBE1A4
0	9	9	-3640557	-757363	-235345	0	2478264	2713609	1650313	2407676	1654330	5294887	0xDDA4A52435265197
0	18729	18729	24220434	-1652632	-203289	-5325	8204526	8407815	50045378	51698010	109420843	85200409	0x0A45CBD850200611
0	43404	43404	5963941	1332979	-169461	-5492	1092898	1262359	22851019	21518040	28891675	22927734	0x89F23668C40B40AB
0	180051	180051	61581	25652	-109752	587	989575	1099327	32235129	32209477	32334203	32272622	0x5C87FDC013E447DF

ANALYSIS OF WORKLOAD-TEST-RESULTS PERSONAL SUMMARY

Tool	Ease of Use	Flexibility	Level of Detail
WorkloadViewer	3	1	1
Power BI Dashboard	2	2	1
Query Store Script	1	3	3

Community-Spirit



RE: WorkloadTools stuck after replaying for 2.4 hours

G gianluca.sartori@sqlconsulting.it
An Guth, Martin, 3C Deutschland GmbH

Antworten | Allen antworten | Weiterleiten | ...

Do 30.04.2020 18:24

Sie haben am 30.04.2020 18:48 auf diese Nachricht geantwortet.

Hi Martin,

I'm sorry that you're having trouble with WorkloadTools.
Unfortunately, without any error message I cannot tell what is happening in the replay. Do you have any error messages to share from your logs?

I'm sorry for no sequence_number getting collected. Turns out this error applies only to XE streaming, so I added this property and now it should be collected correctly in 1.5.7.
For the time being you could update event_sequence with row_id and you should be fine.

In the past I used WT to capture and replay for days or even weeks, so I have no idea what is going wrong in your case. I have to say that in those case I used it with the realtime replay and I have never captured a 6.32 GB sqlite file. However, it should not make a difference.
Synchronization mode again should not be the source of the problem.

I just relased v 1.5.7 and I hope it addresses some of your points.
Good luck with your project!

Cheers
Gianluca

From: Guth, Martin, 3C Deutschland GmbH
Sent: mercoledì 29 aprile 2020 17:59
To: gianluca.sartori@sqlconsulting.it
Subject: WorkloadTools stuck after replaying for 2.4 hours

Hi Gianluca,

I was quite motivated after seeing a 30 minutes replay finish in 30 minutes and therefore tried a longer replay.
I recorded a workload for 28 hours and started the replay.
However after approx.. 2.4 hours (9%) into the replay it suddenly stucks.
Sp_whoIsActive shows no activity and the console doesn't show any additional messages for hours....to me it seems that the replay just died.

Bugfix und neues Release von Gianluca bereits nach einem Tag

DEMOS

TIPPS & TRICKS

Eine Logging-Tabelle ist super hilfreich

- Falls noch keine Logging-Tabelle: Anlage einer Tabelle mit mindestens einer DateTime Spalte
- Hinzufügen eines neuen Datensatzes jede Minute während Workloadaufnahme z.B. über Agent-Job
- Beispiel:

Results | Messages

	zeitpunkt
1	2020-05-09 11:39:00.290
2	2020-05-09 11:40:00.290
3	2020-05-09 11:41:00.290
4	2020-05-09 11:42:00.290
5	2020-05-09 11:43:00.290

- Einsatzzweck: Monitoring der Replaygeschwindigkeit

Restore optimieren

- Point-In-Time Restore mit Log-Backups
  - Zum Start der Aufzeichnung eine Marked Transaction ausführen

```
1 BEGIN TRANSACTION CaptureStart WITH MARK 'CaptureStart';
2 SELECT GETDATE();
3 COMMIT TRANSACTION CaptureStart;
```

  - Restore der Logbackups mit Angabe von WITH STOPATMARK = 'CaptureStart'
  - Alternativ mit Angabe von WITH STOPAT und dem Startzeitpunkt
- Datenbanksnapshot nach erstem Restore für schnellere Folge-Restores (ab SQL 2016 auch in Standard Edition ☺)

LESSONS LEARNED

Lessons learned Workloadtests

- Komplexes Thema → reserviert euch ausreichend Zeit
- In der „eigenen“ DB gibt es Interessantes im Workload zu entdecken (z.B. Zugriffsversuche auf Tabellen, die bereits vor Monaten gelöscht wurden)
- Manchmal ergibt es Sinn, den Workload zu manipulieren
- Bedenken, dass alles aufgezeichnet wird, was in der DB passiert
    - Achtung bei Backup-Befehlen → mindestens sicherstellen, dass keine Backups der Produktion überschrieben werden

Lessons learned PASS-Vortrag

- Weniger ist mehr: besser kritisch hinterfragen, wie viel man präsentiert
- Demos brauchen mehr Vorbereitungszeit als gedacht
- Erstellung eines Workloads ist aufwändiger als gedacht

Ressourcen

- https://github.com/spaghettidba/WorkloadTools
- https://spaghettidba.com/
- https://sqlitebrowser.org/
- https://www.brentozar.com/archive/2019/04/free-sql-server-load-testing-tools/
  https://www.brentozar.com/archive/2019/01/how-to-load-test-a-database-application/

FRAGEN UND DISKUSSION