PASS SUMMIT 2016

Lockless in Seattle

Using In-Memory OLTP for Transaction Processing

Mark Broadbent, Principal,
SQLCloud

PASS

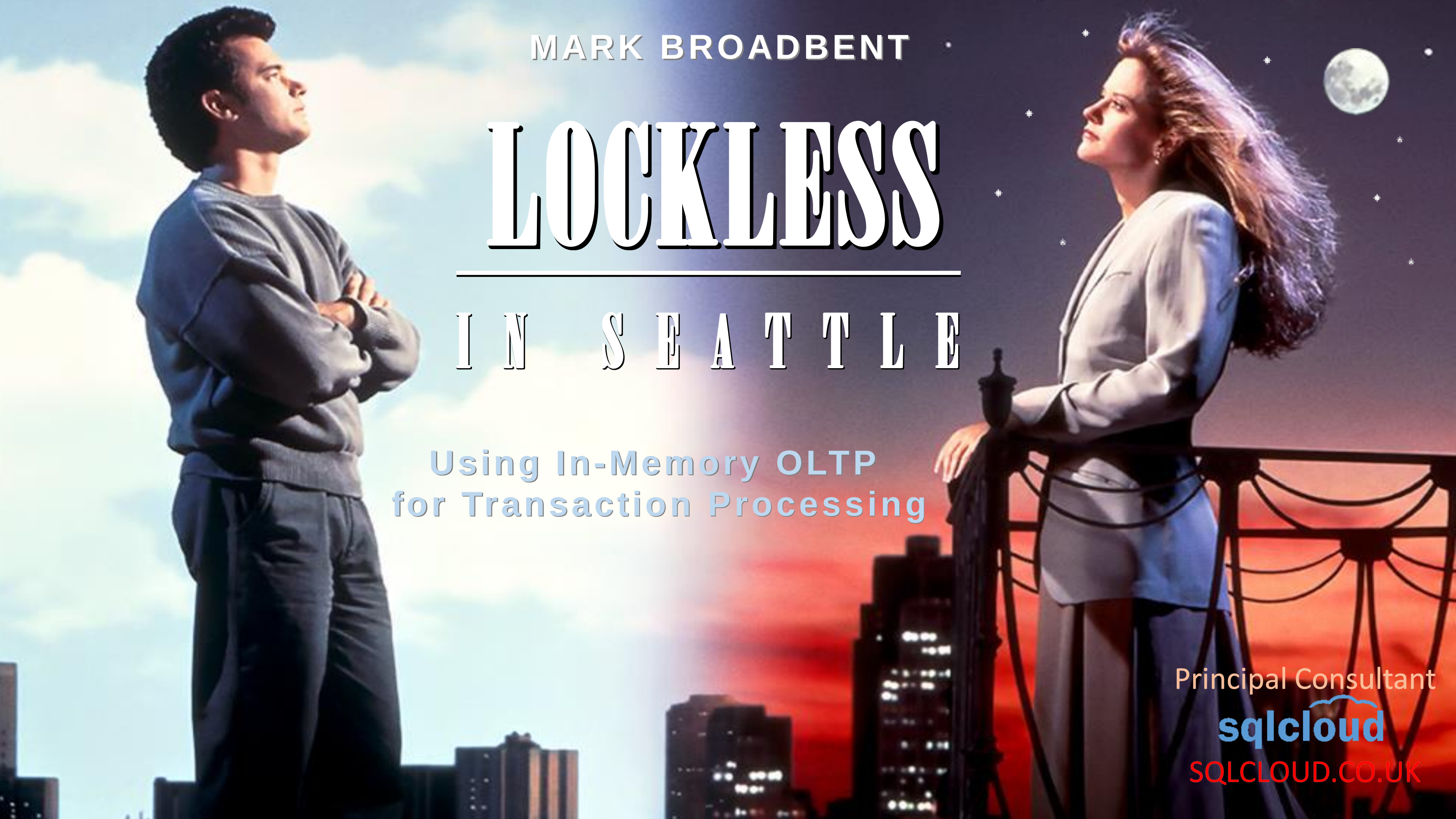
Please silence cell phones

MARK BROADBENT

LOCKLESS

IN SEATTLE

Using In-Memory OLTP for Transaction Processing



Principal Consultant

sqlcloud

SQLCLOUD.CO.UK

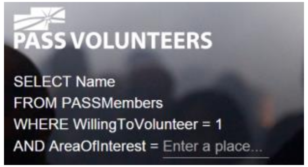
Explore Everything PASS Has to Offer

24 HOURS OF PASS

FREE ONLINE WEBINAR EVENTS

PASS SQL saturday

FREE 1-DAY LOCAL TRAINING EVENTS



PASS VOLUNTEERS

SELECT Name
FROM PASSMembers
WHERE WillingToVolunteer = 1
AND AreaOfInterest = Enter a place...

VOLUNTEERING OPPORTUNITIES

PASS CHAPTERS

LOCAL USER GROUPS AROUND THE WORLD

PASS VIRTUAL CHAPTERS

ONLINE SPECIAL INTEREST USER GROUPS

PASS CommunityCONNECTOR

PASS COMMUNITY NEWSLETTER

PASS BLOG

WHITE PAPERS

SESSION RECORDINGS

FREE ONLINE RESOURCES

PASS BUSINESS ANALYTICS CONFERENCE 2016

BUSINESS ANALYTICS TRAINING

PASS BUSINESS ANALYTICS CONFERENCE 2016

Data. Insights. Action

Your gateway to Business Analytics Community news

PASS BAC 2016 Registration Is Open!

Call for Speakers—Apply for PASS BAC 2016 Now!

Training Available Through PASS

BA INSIGHTS NEWSLETTER

PASS VIRTUAL CHAPTERS

www.sqlpass.org/vc

PASS APPLICATION DEVELOPMENT VIRTUAL CHAPTER

PASS BIG DATA VIRTUAL CHAPTER

PASS BUSINESS ANALYTICS VIRTUAL CHAPTER

PASS BUSINESS INTELLIGENCE VIRTUAL CHAPTER

PASS CLOUD VIRTUAL CHAPTER

PASS DATA ARCHITECTURE VIRTUAL CHAPTER

PASS DATABASE ADMINISTRATION VIRTUAL CHAPTER

PASS DATA SCIENCE VIRTUAL CHAPTER

PASS DBA FUNDAMENTALS VIRTUAL CHAPTER dba

PASS EXCEL BI VIRTUAL CHAPTER

PASS GLOBAL CHINESE VIRTUAL CHAPTER

PASS GLOBAL FRANCOPHONE VIRTUAL CHAPTER

PASS GLOBAL HEBREW VIRTUAL CHAPTER עברית

PASS GLOBAL ITALIAN VIRTUAL CHAPTER

PASS GLOBAL PERSIAN VIRTUAL CHAPTER فارسی

PASS GLOBAL PORTUGUESE VIRTUAL CHAPTER PT

PASS GLOBAL RUSSIAN VIRTUAL CHAPTER AЯ

PASS GLOBAL SPANISH VIRTUAL CHAPTER ñ

PASS HEALTHCARE VIRTUAL CHAPTER

PASS HIGH AVAILABILITY & DISASTER RECOVERY VIRTUAL CHAPTER

PASS HYBRID VIRTUAL CHAPTER

PASS INFRASTRUCTURE VIRTUAL CHAPTER

PASS IN MEMORY VIRTUAL CHAPTER

PASS PERFORMANCE VIRTUAL CHAPTER

PASS POWERSHELL VIRTUAL CHAPTER

PASS PROFESSIONAL DEVELOPMENT VIRTUAL CHAPTER

PASS SATURDAY NIGHT SQL VIRTUAL CHAPTER

PASS SECURITY VIRTUAL CHAPTER

PASS VIRTUALIZATION VIRTUAL CHAPTER

PASS WOMEN IN TECHNOLOGY VIRTUAL CHAPTER W.I.T

Session Evaluations

Your feedback is important and valuable.

3

ways to access

Submit by 5pm Friday November 6th to **WIN** prizes

Go to **passSummit.com**

Download the GuideBook App and search: **PASS Summit 2016**

Follow the QR code link displayed on session signage throughout the conference venue and in the program guide

Contact…

mark.broadbent@sqlcambs.org.uk

@retracement

tenbulls.co.uk

Likes…

--- E L I T E ---

Neuen Commander laden (J/N)?

(C) D.Braben & I.Bell 1985

Guilty pleasures…

docker

Badges…

Microsoft CERTIFIED
Master: SQL Server

MVP Microsoft® Most Valuable Professional

Microsoft® Community Contributor AWARD

Microsoft CERTIFIED Solutions Expert
Data Platform

Microsoft CERTIFIED Solutions Expert
Data Management and Analytics

PASS
OUTSTANDING PASS VOLUNTEER

Community…

PASS HYBRID VIRTUAL CHAPTER

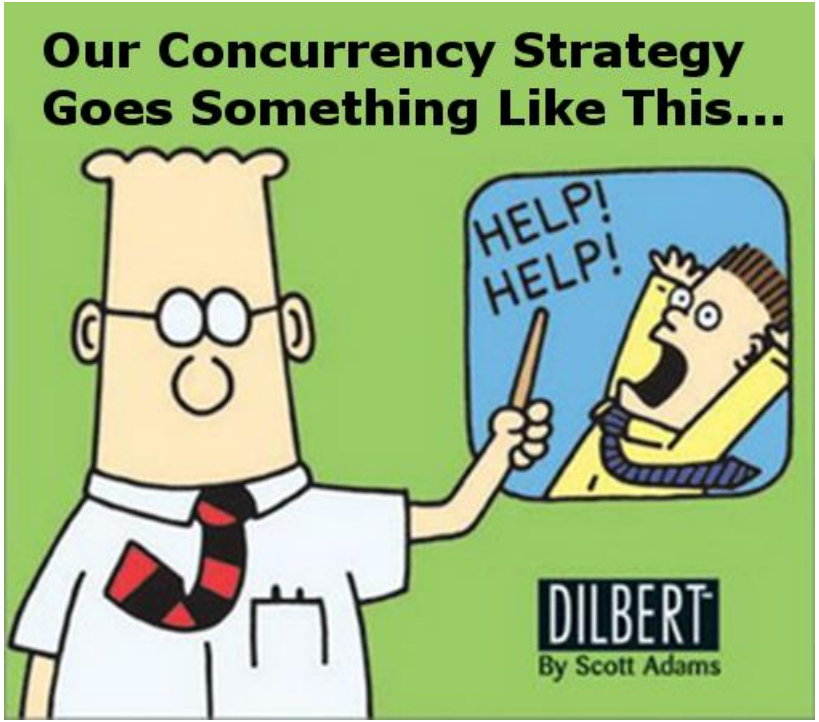
SQLEA

SQL SATURDAY CAMBRIDGE

SHAREPOINT SATURDAY CAMBRIDGE

SQL Nexus

Agenda



1 Why IMOLTP?

2 Architecture

3 Implementation

4 Isolation & TP Control

4 Limitations

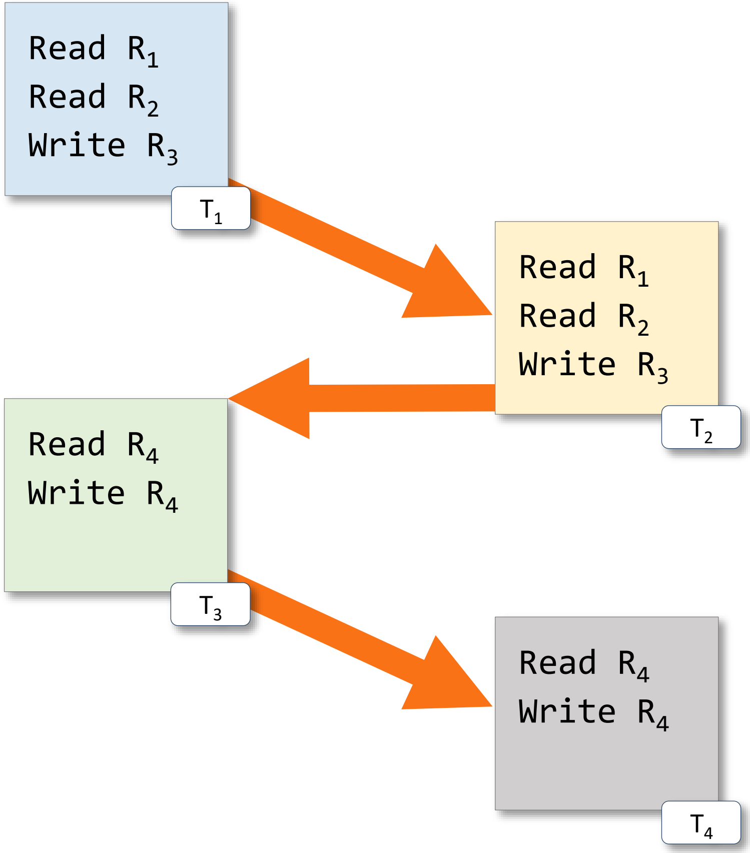
5 The best part of the presentation... Gin O‘Clock

Serial Processing

- Theoretically at least, serial execution time (should) = our slowest throughout speed.

  (This is the 2nd law of Concurrency Control)

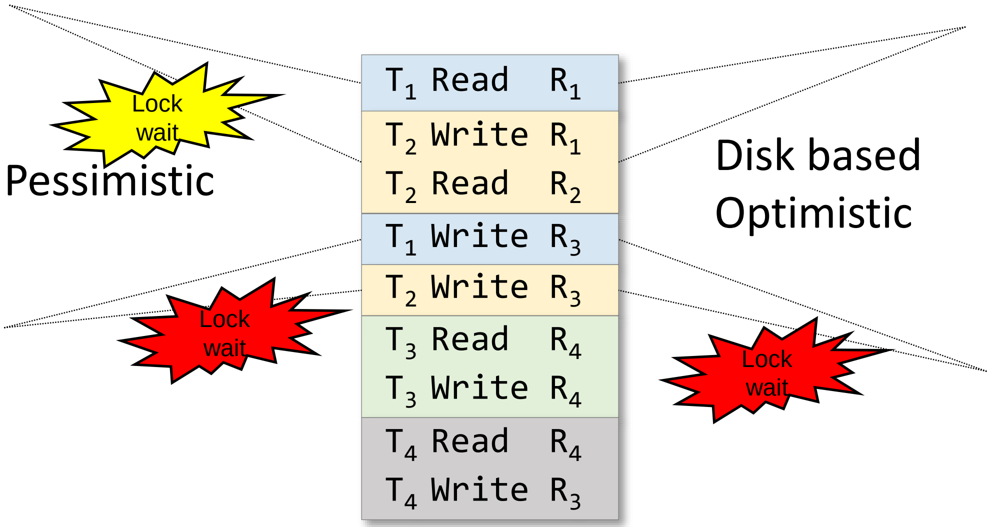
- Our aim then is to execute workloads in parallel right?!
- But when we do, our typical resource bottlenecks are:
  - Memory
  - CPU
  - Disk IO



Parallel Processing requires Transaction Interleaving

T_2 (write) waits for T_1 to release Slock on R_1
Slock is released when?

T_2 (write) waits for T_1 to release Xlock on R_3
Xlock is released when?



T_1	Read	R_1
T_2	Write	R_1
T_2	Read	R_2
T_1	Write	R_3
T_2	Write	R_3
T_3	Read	R_4
T_3	Write	R_4
T_4	Read	R_4
T_4	Write	R_3

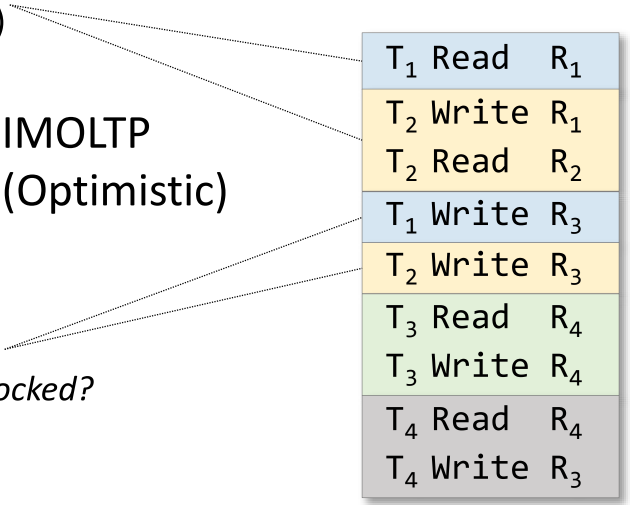
T_2 (write) is not blocked by T_1 (read)
Why no blocking?

T_2 (write) waits for T_1 (write) to release Xlock on R_3
Why is there blocking?

Parallel Processing requires Transaction Interleaving

T_2 (write) will not be blocked by T_1 (read)
(same behaviour on-disk)

IMOLTP
(Optimistic)



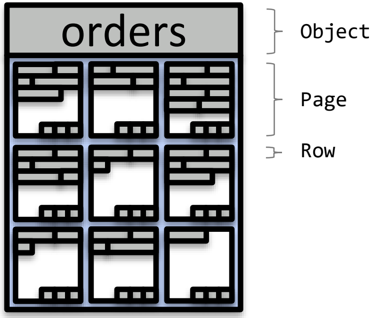
T_1	Read	R_1
T_2	Write	R_1
T_2	Read	R_2
T_1	Write	R_3
T_2	Write	R_3
T_3	Read	R_4
T_3	Write	R_4
T_4	Read	R_4
T_4	Write	R_3

T_2 (write) will not be blocked by T_1 (write)
Why are writes not blocked?

Concurrency Models in SQL 2014 and beyond

- Pessimistic Isolation
  - Readers do not block readers
  - Writers block readers
  - Readers block writers
  - Writers block writers
- (disk based) Optimistic Isolation
  - Readers do not block readers
  - Writers do not block readers
  - Readers do not block writers
  - Writers block writers
- (In-Memory) Optimistic Isolation
  - Readers do not block readers
  - Writers do not block readers
  - Readers do not block writers
  - *Writers do not block writers*

Governed by lock compatibility and implemented through lock hierarchy and escalation

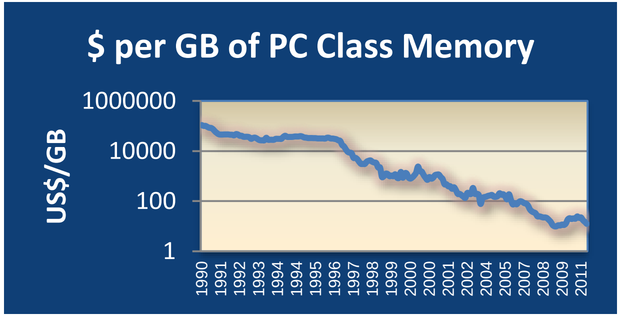


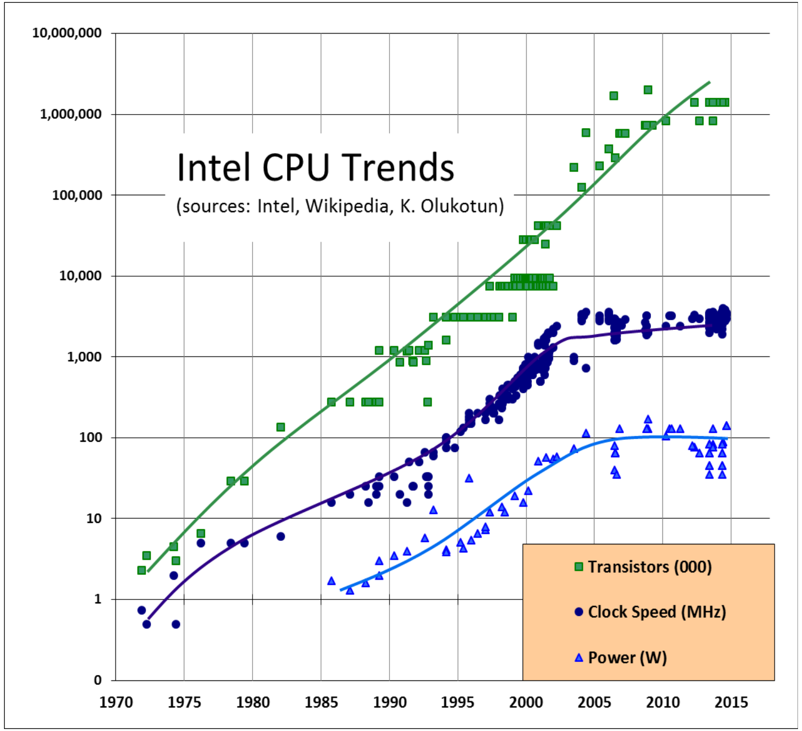
Sch-M, or object level X can kill concurrency. Intent locks and escalation have overhead!

Governed by write conflict detection

Hardware Trends

- Hardware trend is for scale not speed
- CPU Core count increases, clock speed static
- Memory sizes increase/ costs fall
- (As disk speeds also increase)
- Concurrency is clearly a software problem

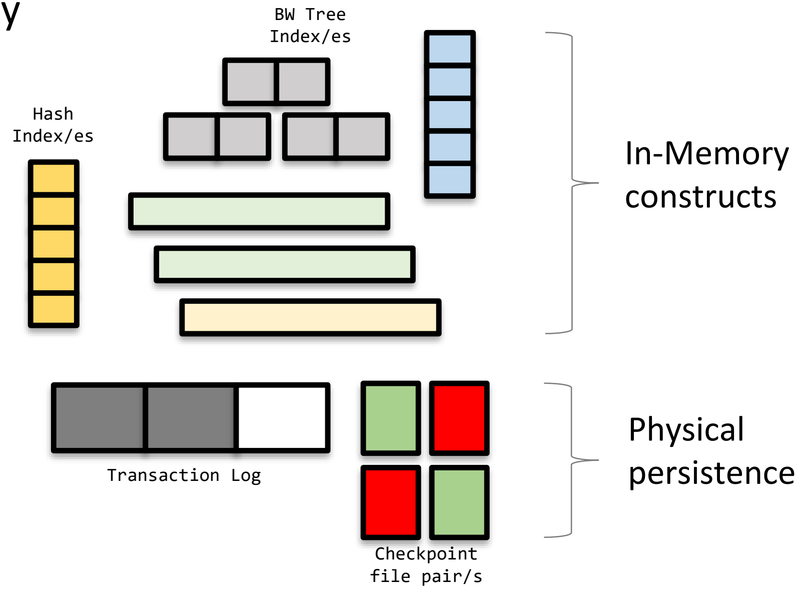




Graphics from BRK3576, In-Memory – The Road Ahead by Kevin Farlee – Ignite Conference 2015

In-Memory OLTP to the rescue…

- In-Memory data structures (optimised for memory)
- Persistence through Transaction Log and checkpoint file pair/s
- Logging optimizations and improvements
- No TempDB overhead – all versioning In-Memory
- Lockless and latchless operation
- No fragmentation concerns
- Baked into product

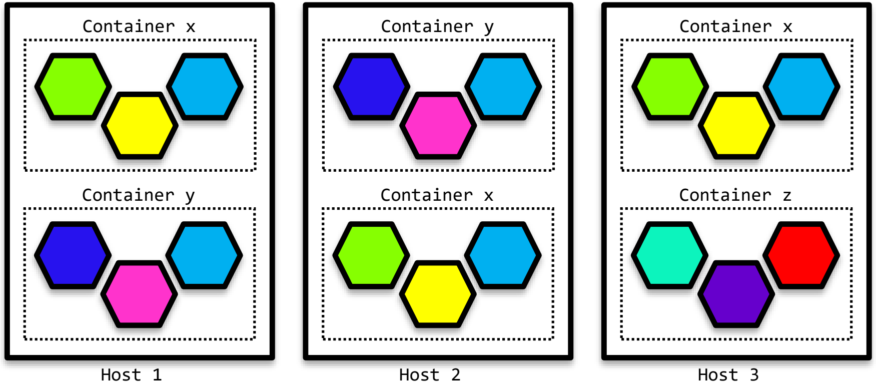


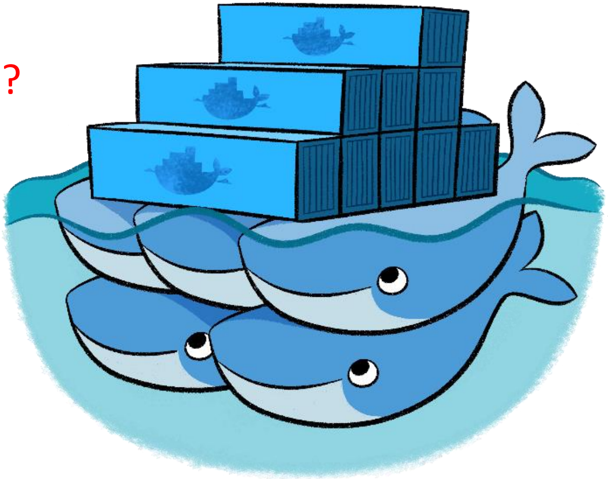
A solution to a problem that doesn’t exist?

- Before Internet we were in db-centric world. We are now in a message centric world
- A defined platform tends to define a solution, when in fact it should be the problem driving the solution...
- Micro-services, stateless data queues, smaller data repositories, no locking
- Application Caching Fabrics reduce latencies

Newsflash so does SQL Server Caching/ Buffering

- Are monolithic OLTP environments about to become extinct?

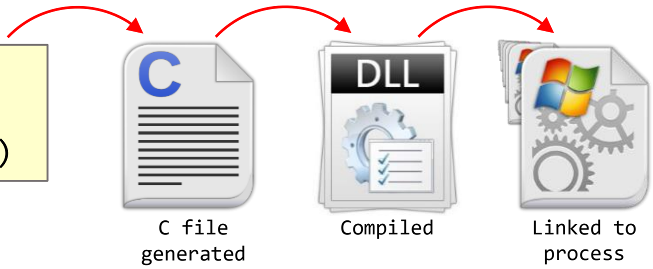




In-Memory Data Structures (Tables and Indexes)

- Tables
  - On create - C file generated, Compiled to DLL and Linked to SQL process
  - Structure of the row is optimized for memory residency and access
  - Recreated, compiled, linked on database startup
- Indexes (rebuilt at startup)
  - Hash indexes for point lookups
  - Memory-optimized non-clustered index for range and ordered scans
  - Do not duplicate data, just pointers on rows
  - Warning! Statistics not auto-updated.

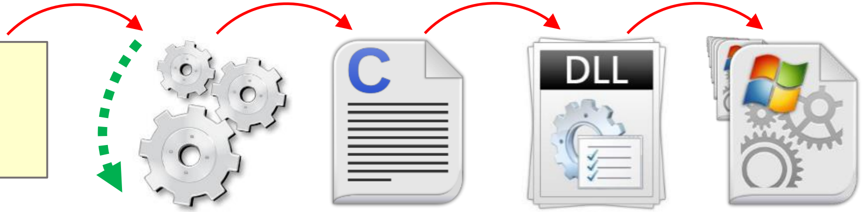
```
CREATE TABLE…
WITH
(MEMORY_OPTIMIZED=ON)
```



In-Memory Data Structures (Natively Compiled Stored Procedures)

- On create - Optimized, C file generated, Compiled to DLL and Linked to SQL process
- Warning, optimised once and Plan based on statistics!
- Not part of plan cache (so not visible in `sys.dm_exec_cached_plans`)
- Recreated on database startup and compiled on first execution
- They can only access In-Memory Tables

```
CREATE PROCEDURE…
WITH
NATIVE COMPILATION
```



Query Optimization

C file generated

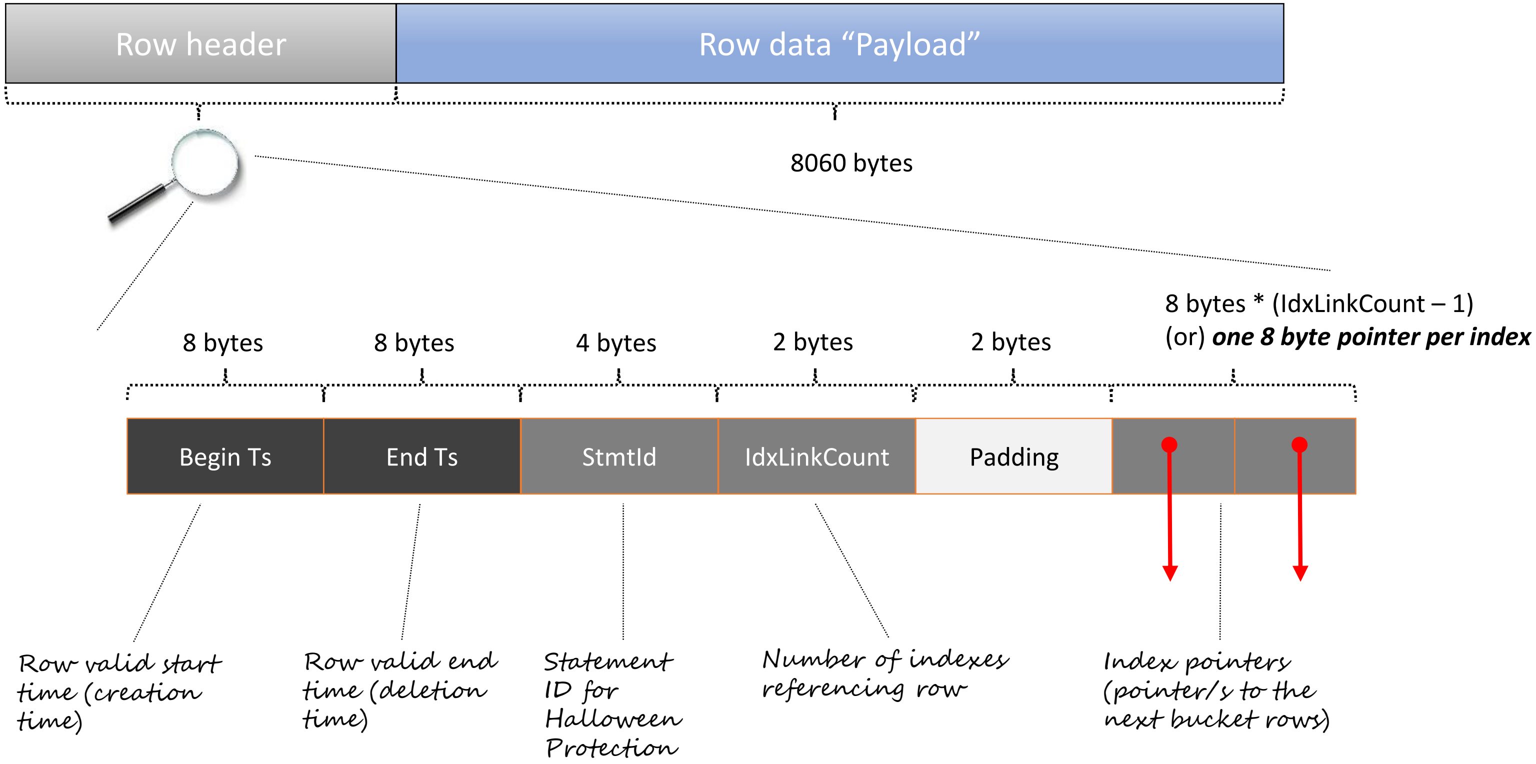
Compiled

Linked to process

Demo

Creating in memory tables

MemOpt Table: Row Format



Row header

Row data “Payload”

8060 bytes

8 bytes

8 bytes

4 bytes

2 bytes

2 bytes

8 bytes * (IdxLinkCount – 1)
(or) ***one 8 byte pointer per index***

Begin Ts

End Ts

StmtId

IdxLinkCount

Padding

Row valid start time (creation time)

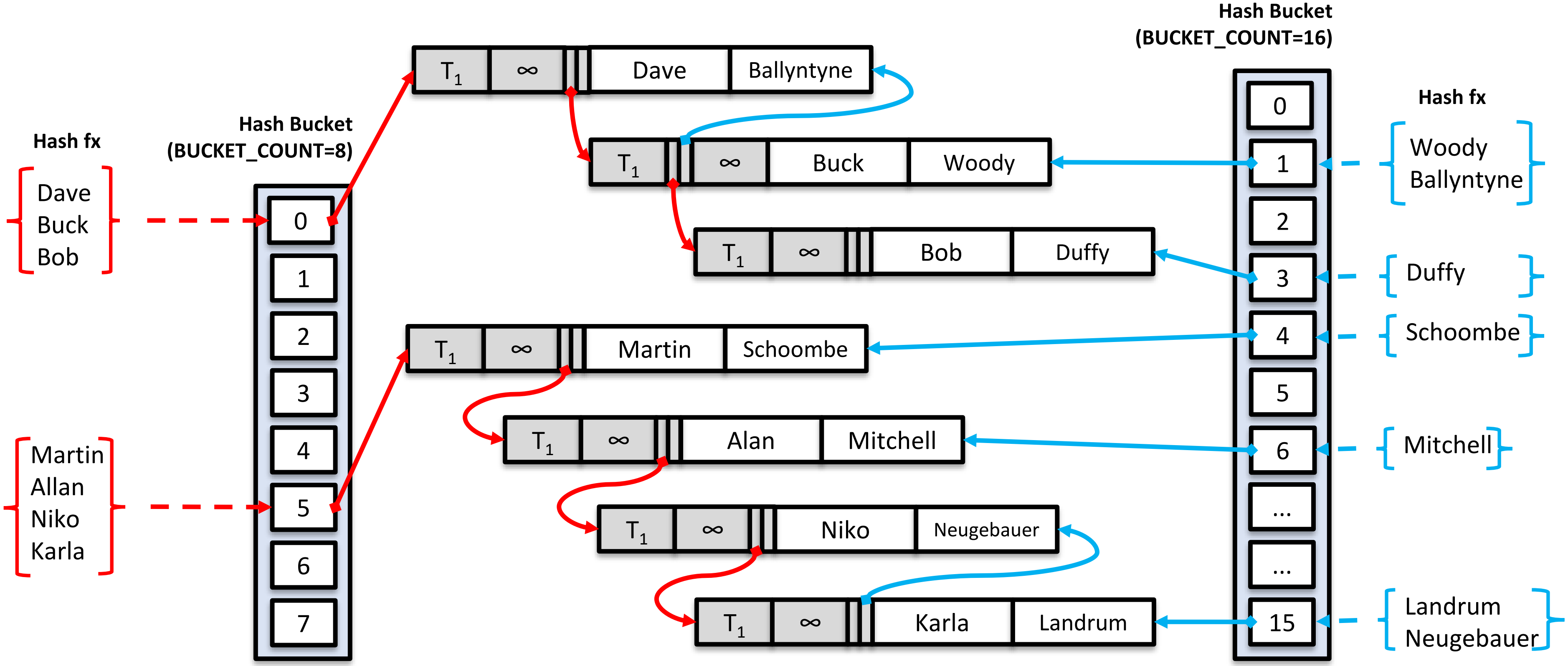
Row valid end time (deletion time)

Statement ID for Halloween Protection

Number of indexes referencing row

Index pointers (pointer/s to the next bucket rows)

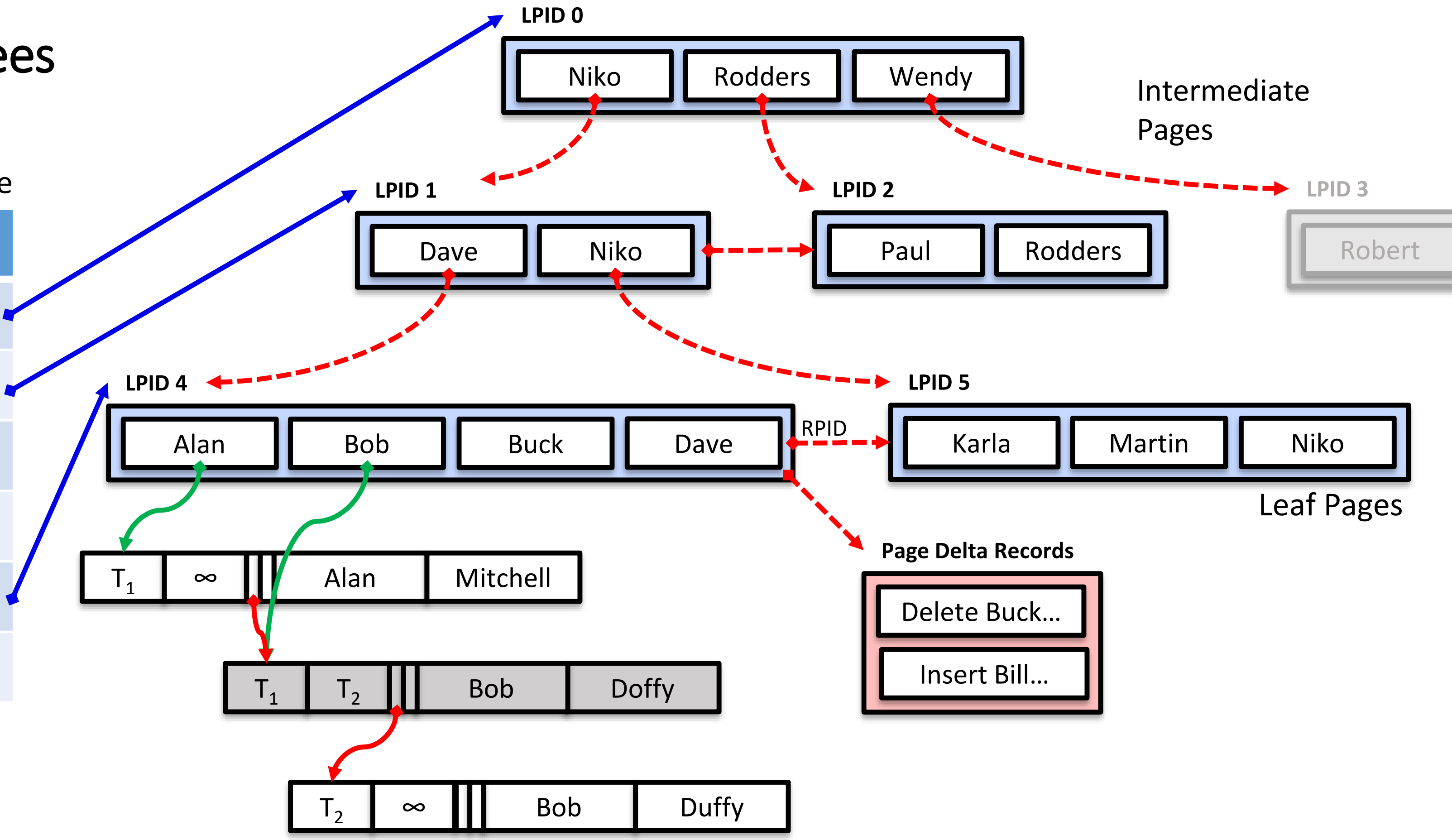
Hash Indexes



BW-Trees

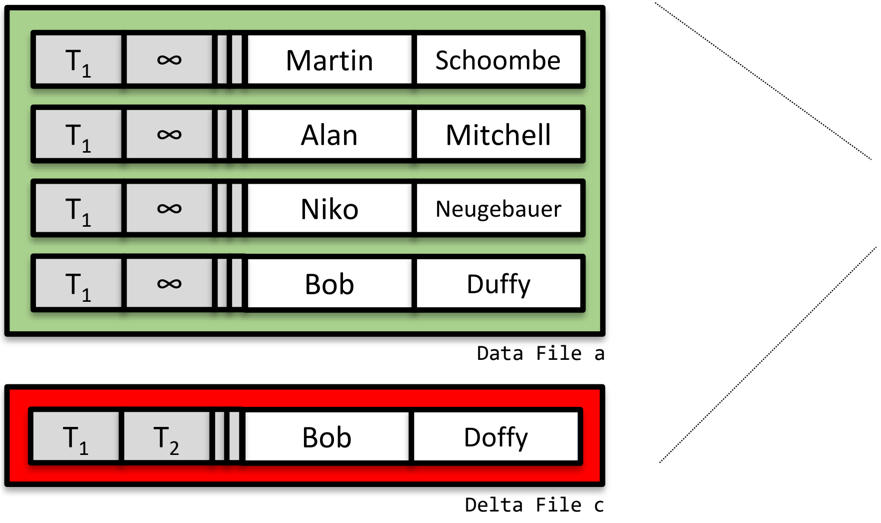
Page mapping table

LPID	PMA
0	0x00008862
1	0x00009862
2	0x00008AA2
3	0x000077C6
4	0x00005B62
5	0x000088BC



Data Persistence

- Checkpoint file pairs, minimum of 8 (in various states)
- Log file primary persistence source
- Database recovery from CFP and Log
- INS to data file, DEL to delta file, Update is INS and DEL

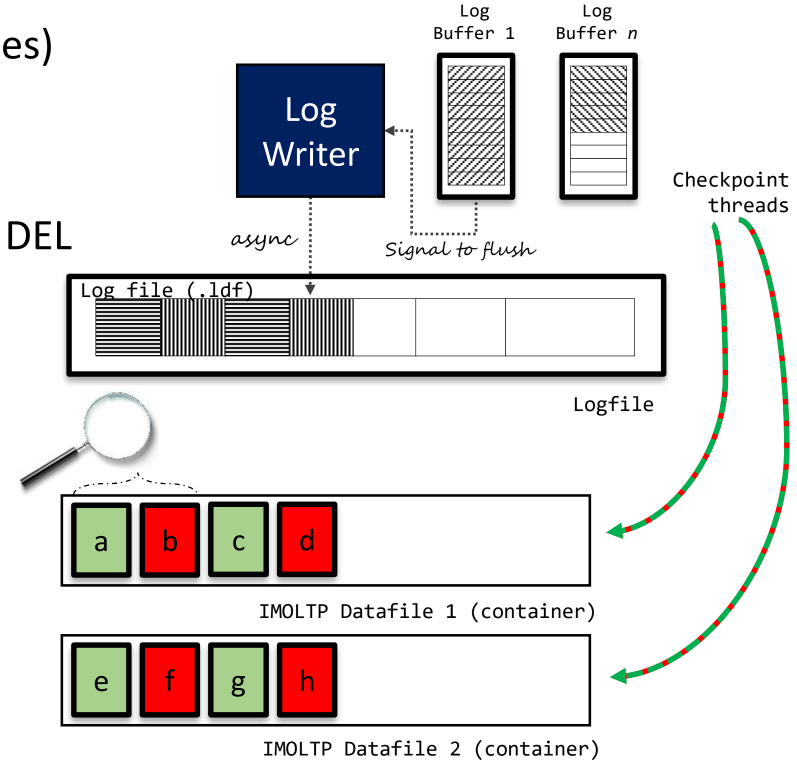


T_1	∞		Martin	Schoombe
T_1	∞		Alan	Mitchell
T_1	∞		Niko	Neugebauer
T_1	∞		Bob	Duffy

Data File a

T_1	T_2		Bob	Doffy

Delta File c



Log Buffer 1

Log Buffer *n*

Log Writer

async

Signal to flush

Checkpoint threads

Log file (.ldf)

Logfile

a b c d

IMOLTP Datafile 1 (container)

e f g h

IMOLTP Datafile 2 (container)

Log Performance improvements

- Indexes not persisted, rebuilt on start-up
  - So NO index maintenance logging
- Log records ordering by Transaction End Timestamps (On disk ordered by LSN)
  - Removes requirement for single log stream to disk per DB (implemented in SQL 2016)
- Transaction consolidation into reduced number of log records
  - Because only committed transactions
  - So NO undo overhead!
- NVDIMM support in Windows 2016!
  - See Accelerating SQL Server 2016 peformance with Persistent Memory in Windows Server 2016

Demo

Logging and improvements

Supported Isolation Levels when accessing IMOLTP tables

		Isolation Level	"Bad Dependencies"	
*1	1	~~READ UNCOMMITTED~~	Dirty Read, Non-Repeatable Read, Phantoms, Lost Updates	
	2	~~READ COMMITTED~~	Non-Repeatable Read, Phantoms , Lost Updates	*2

*1 Politely elevated (mapped to SNAPSHOT) through MEMORY_OPTIMIZED_ELEVATE_TO_SNAPSHOT = ON
*2 But is supported for auto-commit transactions

3	REPEATABLE READ	Phantoms
4	SERIALIZABLE	None
5	SNAPSHOT	*None (though Causal consistency concerns, lost update prevention and other behaviours)*

Querying data

- Interop for:
  - Cross container queries (but introduce synchronisation concerns)
  - Best support
  - But uses legacy engine and incurs unnecessary overhead
- Native Compilation for:
  - Best Performance (but…)
  - Is the most restrictive (and cannot query non in-memory tables)

Cross-Container Transactions

- Cross container transactions therefore are really two internal transactions that are synchronized together.
- Cross container (disk/ in-memory) table transactions are supported but only for:
  - READCOMMITTED + in-memory SNAPSHOT
  - READCOMMITTED + in-memory REPEATABLEREAD/ SERIALIZABLE
  - REPEATABLEREAD/ SERIALIZABLE + in-memory SNAPSHOT
- Are synchronization issues for:
  - SNAPSHOT + ANY In-Memory isolation (so cant do it!)

Demo

Isolation

Gotchas

- Not enough memory to load table causes IMOLTP database stuck in recovery $*_1$
- Running out of memory no transactions
- Combined size of ACTIVE Checkpoint File Pairs on disk equates to approximately 2x the size of table that's in Memory (depending upon frequency of updates)
- In-memory row versions and maintenance operations can require in excess of 2-3x size of table in Memory
- Long running transactions consume more memory for in-memory versions
- No data overwritten on disk. Sequential IO is KEY so locate containers on drives optimized for this
- Disk is even more important because:
    - Transaction log is still king for transaction speed
    - Log truncation dependant on data persistence!

$*_1$ Think about database restores

In-Memory OLTP Limitations

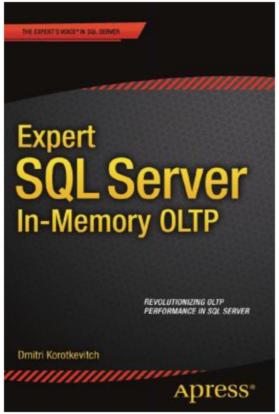
Addresses	Feature/Limit	SQL Server 2014	SQL Server 2016
Performance	Parallelism	Not supported	Supported
Performance	Maximum combined size of durable tables	256 GB	~~2 TB~~ Physical bounds
Performance	Offline Checkpoint Threads	1	1 per container
Performance/ Management	ALTER PROCEDURE / sp_recompile	Not supported	Supported (fully online)
Performance/ Management	ALTER TABLE	Not supported, (DROP / re-CREATE)	Partially supported, (offline operation)
Performance/ Compatibility	Nested native procedure calls	Not supported	Supported
Performance/ Compatibility	Natively-compiled scalar UDFs	Not supported	Supported
Performance/ Compatibility	Indexes on NULLable columns	Not supported	Supported
Compatibility	LOB (varbinary(max), [n]varchar(max))	Not supported	Supported
Compatibility	DML triggers	Not supported	Partially supported (AFTER, natively compiled)
Compatibility	Non-BIN2 collations in index key columns	Not supported	Supported
Compatibility	Non-Latin codepages for [var]char columns	Not supported	Supported
Compatibility	Non-BIN2 comparison / sorting in native modules	Not supported	Supported
Integrity/ Compatibility	Foreign Keys	Not supported	Supported
Integrity/ Compatibility	Check/Unique Constraints	Not supported	Supported
Compatibility	OUTER JOIN, OR, NOT, UNION [ALL], DISTINCT, EXISTS, IN	Not supported	Supported
Compatibility	Multiple Active Result Sets	Not supported	Supported
Management	SSMS Table Designer	Not supported	Supported
Security	Transparent Data Encryption (TDE)	Not supported	Supported

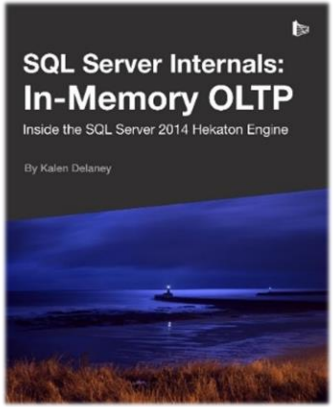
2014 Unsupported features https://msdn.microsoft.com/en-us/library/dn246937(v=sql.120).aspx
2016 Unsupported features https://msdn.microsoft.com/en-us/library/dn246937(v=sql.130).aspx

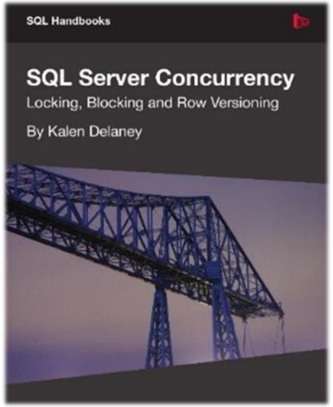
List based from original post http://bit.ly/2euFIxz at SQLPerformance.com written by Aaron Bertrand

Further reading (research papers and books)

- In-Memory OLTP (In-Memory Optimization) https://msdn.microsoft.com/en-us/library/dn133186.aspx
- High-Performance Concurrency Control Mechanisms for Main-Memory Databases - Per-Åke Larson, Spyros Blanas, Cristian Diaconu, Craig Freedman, Jignesh M. Patel, Mike Zwilling
- The Hekaton Memory-Optimized OLTP Engine - Per-Ake Larson, Mike Zwilling, Kevin Farlee
- Concurrent Programming Without Locks – Keir Fraser, Tim Harris
- The Bw-Tree: A B-tree for New Hardware Platforms – Justin J. Levandoski, David B. Lomet, Sudipta Sengupta


THE EXPERT'S VOICE® IN SQL SERVER
Expert
SQL Server
In-Memory OLTP
REVOLUTIONIZING OLTP PERFORMANCE IN SQL SERVER
Dmitri Korotkevitch
Apress®





In Summary (what we have learnt today)…

- Rise in CPU cores, abundance of memory and out of date concurrency model is the reason why in-memory OLTP is the future
- Interleaving and concurrent execution is why we hit severe bottlenecks in pessimistic workloads i.e. Blocking is almost assured
- Snapshot Isolation has no bad dependencies, so is a good fit for our new model (as the new default)
- Adoption will be slow due to some of the complexities (or differences) and restrictions (which are being removed). It is both a development and administrative concern
- Microsoft are 110% committed to this technology

Session Evaluations

Your feedback is important and valuable.

3

ways to access

Submit by 5pm Friday November 6th to **WIN** prizes

Go to **passSummit.com**

Download the GuideBook App and search: **PASS Summit 2016**

Follow the QR code link displayed on session signage throughout the conference venue and in the program guide

PASS

Thank you for listening!

Email: mark.broadbent@sqlcambs.org.uk

Twitter: retracement

Blog: http://tenbulls.co.uk

Slideshare: http://www.slideshare.net/retracement