Progress®

OpenEdge Database Performance Innovations

Faster Backups, Smarter Diagnostics, Better Throughput

Soofia Shaik

Principal QA Engineer, OpenEdge Database

Oct 5, 2026

Agenda

- The Challenge: Shrinking Maintenance Windows
- PROBKUP: Multithreaded Backup, Now Default
- PROUTIL IDXCHECK: Faster Offline Health Checks
- PROUTIL IDXCOMPACT: Smarter Index Compaction
- Statement Caching & Diagnostic Data Collection
- Key Takeaways and Q&A

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.



A backup that runs too long stretches your maintenance window, a slow index check delays your recovery strategy, and database downtime is never free.

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

OpenEdge 13.0 Database Utilities Enhancements

- Databases keep growing, and larger databases mean longer backup, restore and maintenance operations
- OpenEdge 13.0 focuses squarely on performance and scalability of core database utilities
- Goal: shorten outages for backup, repair and index maintenance without changing your operational scripts

PROBKUP: Multithreaded **Backup Is Now the Default**

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

How Offline Backup Traditionally Worked

- Single-user process
  - No CRUD operations/Backup queue
- Database blocks backed up sequentially
  - Per area type (Type I/Type II)
  - Per storage area number
- Sequential, single-threaded design meant backup time scaled directly with database size

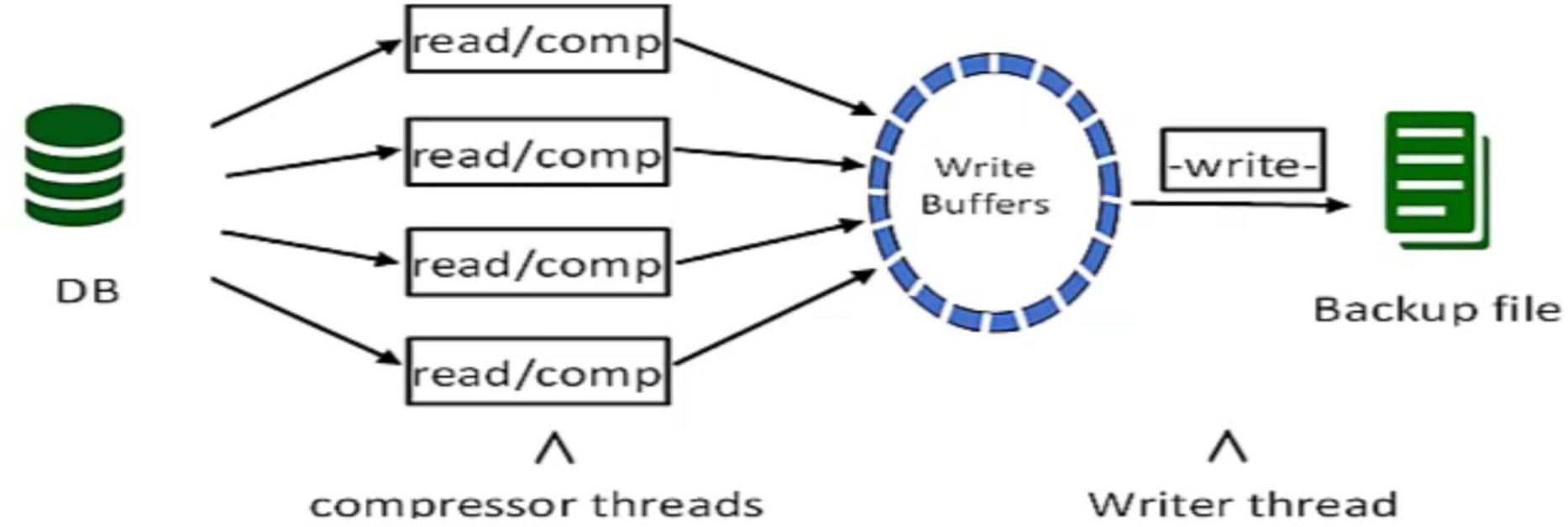
Back Up Offline Databases Faster

- OpenEdge 13.0 introduces multithreaded backup for offline databases & it's now the default for both online AND offline backups
- Improved offline backup process efficiency means less downtime for backup
- Progress recommends backing up an original database before any repair
  - Use cases: back up before repair, back up to create a hot-standby/replication target
- Threaded restore from compressed backup file (OpenEdge 12.7)
  - Up to 2.7 times faster than non-threaded restore
  - Up to 3.7 times faster than non-threaded restore if combined with –noverify option
  - Up to 4.4 times faster than pre-12.5 restore (w/o zstd)

Multithreaded Backup

- Improved Offline Backup Process Efficiency
  - Backup before repair
  - Backup to create hot-standby / replication target
- Less Downtime for Backup

Threaded Backup: Architecture Diagram



Compressor threads read/compress DB blocks in parallel into a ring of write buffers.
A single writer thread flushes them sequentially to the backup file.

Command Syntax & Key Parameters

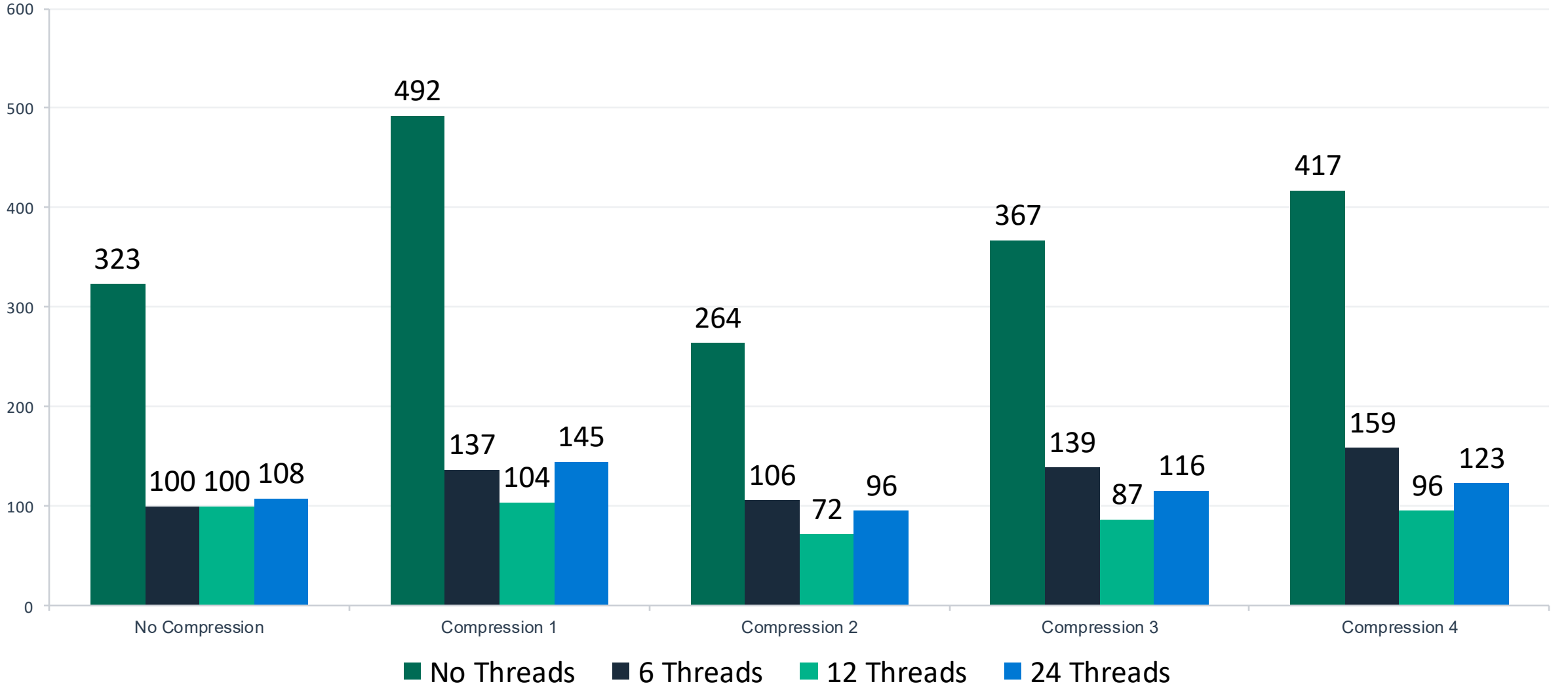
```
probkup db-name device-name [-estimate | -vs n |...]
        [-thread 0|1][-threadnum n][-wbf n]
```

Parameter	Purpose
-thread 0 | 1	Disable/enable multithreaded backup
-threadnum n	Number of worker threads
-wbf n	Number of write buffers

Threaded Offline Backup Performance

86 GB

12 CPU Cores



PROUTIL IDXCHECK Multi-Threaded Index Check Extension

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

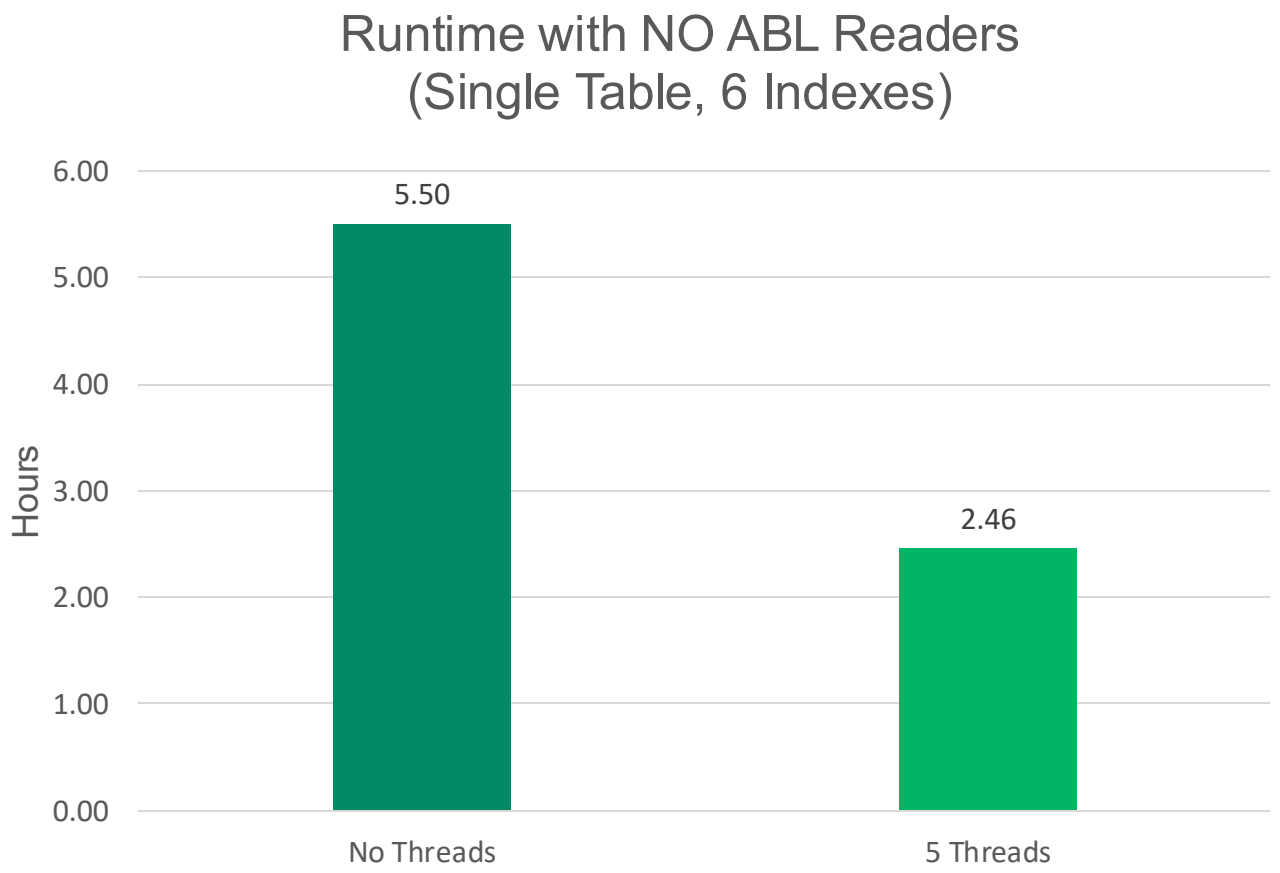
Why It Matters

- Index check is a vital tool for assessing database health, identifying problems and maintaining database integrity.
- It's a cornerstone of your recovery strategy during planned database downtime.
- The bigger the database, the longer a full index check takes—directly extending your outage window.
- OpenEdge 13.0 lets DBAs run index check offline using multiple threads to validate records and key order.

IDXCHECK: Multi-Threading Extended

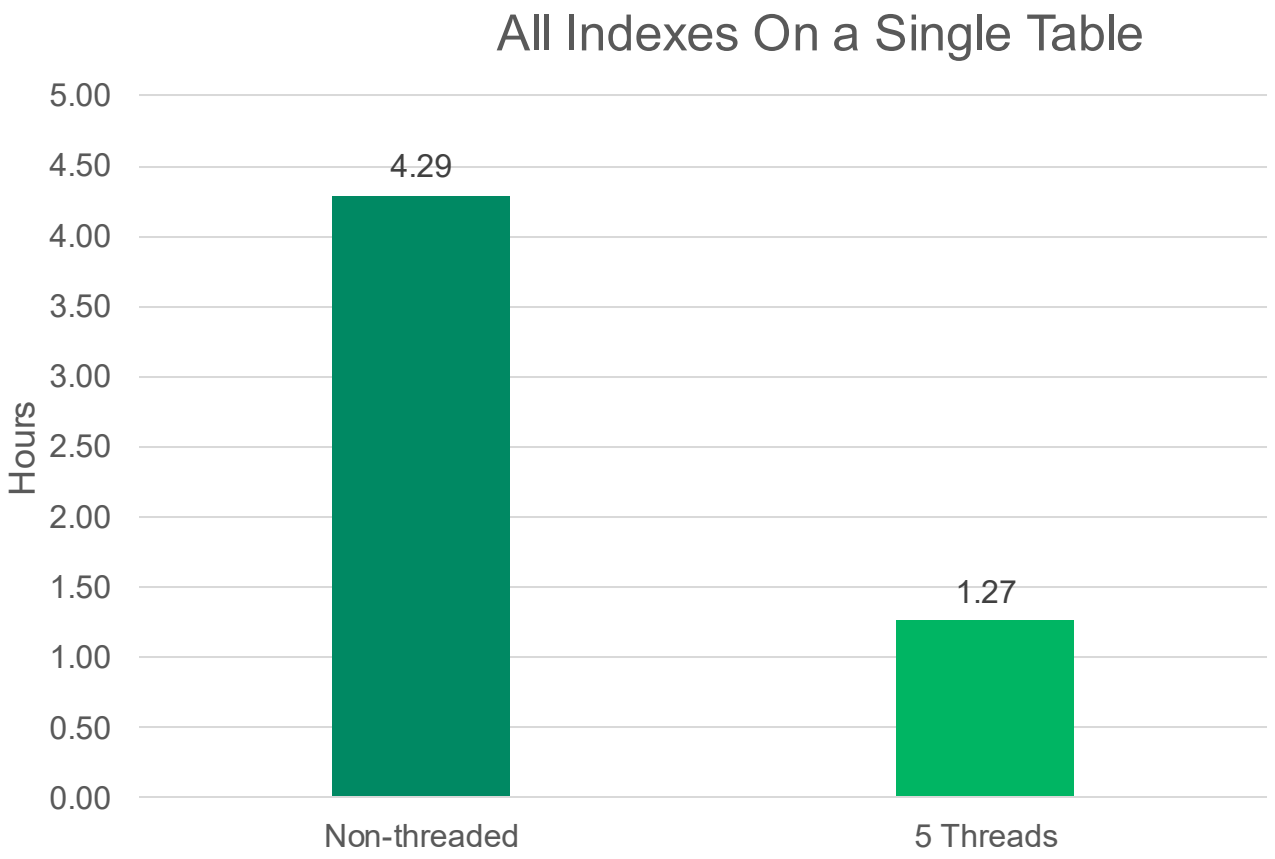
- The OpenEdge platform previously added multithreading to IDXCHECK options 2, 3 and 4 only for online mode.
- OpenEdge 13.0 extends multithreading to offline index check too.
- Same command family: proutil testdb -C idxcheck -thread 1 -threadnum 4
  - -thread turns threading on/off (on by default);
  - -threadnum sets the desired thread/CPU count (default: min (# of CPUs, 6)).
- Performance gain depends on machine workload, database size and layout, indexes and tables.

Multithreaded Online IDXCHECK Performance



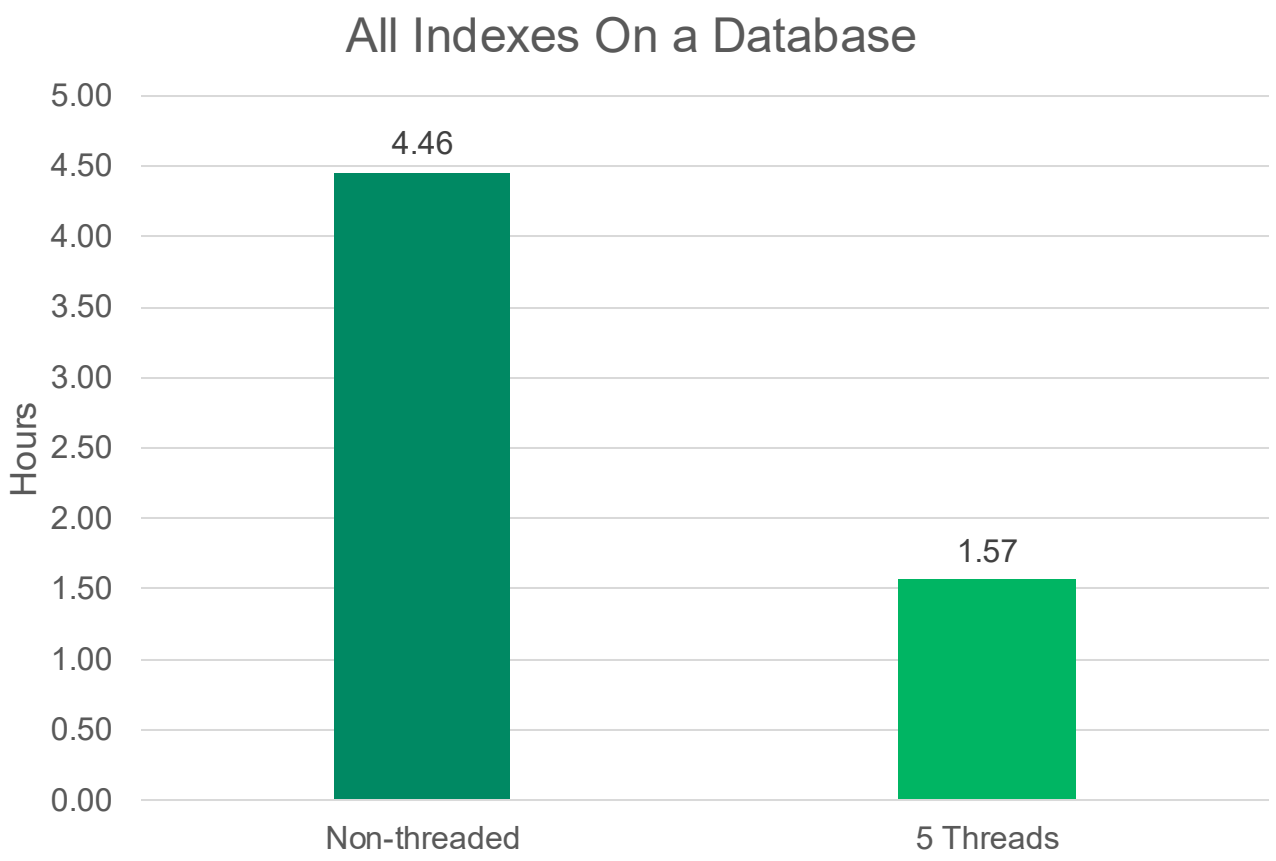
2x faster with threads!

Multithreaded Offline IDXCHECK Performance



3.4x faster with threads!

Multithreaded Offline IDXCHECK Performance



2.8x faster with threads!

PROUTIL IDXCOMPACT Cleaning Up Delete Placeholders, for Good

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

The Problem: Delete Placeholders

- Also known as Delete Locks or Index Locks
- Created during various operations on unique indexes
- Used during subsequent read operations
- Take up storage in the block and count toward utilization percentage
- They can accumulate over time, degrading index performance

Before Index Compact: A Real Example

Database	For Each (ms)	Hit Ratio	Utilization	Blocks	Tree Level	Blocks on Delete Chain	Delete Holders
Original	2131	0.99%	99%	11,182	3	12	9,910,449
Old IDXCOMPACT (95%)	2098	1.01%	99.1%	11,066	3	0	9,818,739

Enhanced IDXCOMPACT: Three New Capabilities

- Even after traditional compaction, nearly 9.9M delete holders remained in this real database - they were never being cleaned up
- OpenEdge 13.0 enhances PROUTIL IDXCOMPACT to finally clean up index delete placeholders
  - Improved status messages shows % compaction completed per iteration/loop, plus estimated time remaining
  - New _UserStatus-State value 40 "Counting blocks for the scan" (visible in PROMON / _UserStatus VST)
  - -compactonly compacts a unique index's tree only, disabling the delete-placeholder scan

IDXCOMPACT: Syntax

- Command syntax
  - proutil dbname -C idxcompact table.index [percentile][unusedblocks][-compactonly]
- The new enhancements are automatic behavior improvements layered on top of the familiar command

Real Results: Delete Holder Cleanup

Database	For Each (ms)	Hit Ratio	Utilization	Number of Blocks	Tree Level	Blocks on Delete Chain	Number of Delete Holders
Original	2131	0.99%	99%	11,182	3	12	9,910,449
Old Index Compact (95%)	2098	1.01%	99.1%	11,066	3	0	9,818,739
New Index Compact (95%)	567	100%	93.9%	123	2	0	0
New Index Compact (50%)	568	100%	50.3%	230	2	0	0

Index Delete Placeholder Detection and Reporting

- 4 new columns will be added to INDEX BLOCK SUMMARY output:
  - % DPsOnChain - bytes occupied by DPs from blocks on index deletechain/bytes occupied in leaf blocks
  - % DPsOffChain - bytes occupied by DPs from blocks not on index deletechain/bytes occupied in leaf blocks
  - LeafBlocks - total number of leaf blocks in the index tree
  - LeafBlocksOnChain - total number of blocks on the delete chain
- -csoutput/-verbose will be updated to display new columns
- Will give users a better understanding of when to use the compaction

Output Updates

```
INDEX BLOCK SUMMARY FOR AREA "IndexArea": 8
-------------------------------------------------------


INDEX BLOCK SUMMARY FOR SHARED OBJECTS:
-----------------------------------------
```

Table	Index	Fields	Levels	Blocks	Size	% Util	Factor	% DPsOnChain	% DPsOffChain	LeafBlocks	LeafBlocksOnChain
PUB.my4gltab1											
myIndex1	8	1	3	70	212.4K	76.4	1.5	0.0	0.0	67	0
myindex2	16	1	4	551	1.5M	68.6	1.6	85.4	0.6	539	469
PUB.my4gltab2											
myIndex1	9	3	4	581	1.3M	58.4	1.8	12.5	35.9	562	72
myindex2	17	3	7	642	1.3M	52.5	1.9	13.9	19.4	572	87
PUB.my4gltab3											
myIndex1	10	5	6	19624	47.0M	61.8	1.8	80.9	2.6	19098	15455
myindex2	18	5	14	25394	58.3M	59.2	1.8	80.3	0.3	15370	12307
PUB.my4gltab4											
myIndex1	11	7	7	20130	48.5M	62.1	1.8	81.3	2.4	19526	15908
myindex2	19	7	14	25746	59.6M	59.7	1.8	80.4	0.3	15370	12303
PUB.MYSQLTAB1											
MYINDEX1	12	1	3	462	1.4M	80.2	1.4	85.7	0.6	458	388
myindex2	20	1	4	551	1.5M	68.6	1.6	85.4	0.6	539	469
PUB.MYSQLTAB2											
MYINDEX1	13	3	4	581	1.3M	58.3	1.8	12.5	27.8	562	72
myindex2	21	3	7	642	1.3M	52.5	1.9	13.5	19.9	572	84
PUB.MYSQLTAB3											
MYINDEX1	14	5	5	19204	46.5M	62.4	1.8	80.9	1.3	18901	15283
myindex2	22	5	14	25391	58.3M	59.2	1.8	80.2	0.4	15360	12280
PUB.MYSQLTAB4											
MYINDEX1	15	7	6	19482	47.4M	62.7	1.7	80.9	1.3	19174	15542
myindex2	23	7	14	25724	59.5M	59.6	1.8	80.2	0.4	15360	12268
Subtotals:				184775	435.1M	60.7	1.8	79.8	1.5	142030	112987

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

-csoutput/-verbose output change

sportsdb.ix.txt

```
 "Area" "Area #" "Partition Type" "Tenancy" "Tenancy #" "Partition Policy" "Partition" "Partition #" "Owner" "Table" "Table #" "Index" "Index #" "State" "Fields" "Level
'Factor" "DPsOnChain" "DPsOffChain" "LeafBlocks" "LeafBlocksOnChain"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "RefCall" 17 "Txt" 52 1 1 1 1 2242   27.5    1.0 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "SalesRep" 18 "SalesRep" 53 1 1 1 1 105    1.3    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "ShipTo" 19 "CustNumShipTo" 54 1 2 1 1 36    0.4    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "State" 20 "State" 55 1 1 1 1 485    5.9    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "Supplier" 21 "SupplierID" 56 1 1 1 1 97    1.2    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "SupplierItemXref" 22 "ItemNumSupplierID" 58 1 2 1 1 624    7.6    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "SupplierItemXref" 22 "SupplieridItemNum" 57 1 2 1 1 542    6.6    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "TimeSheet" 23 "EmpNoDayRecorded" 59 1 2 1 1 293    3.6    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "Vacation" 24 "EmpNoStartDate" 60 1 2 1 1 153    1.9    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "Warehouse" 25 "WarehouseName" 62 1 1 1 1 307    3.8    1.0 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area" 20 "-" "-" 0 "-" "-" 0 "PUB" "Warehouse" 25 "WarehouseNum" 61 1 1 1 1 133    1.6    1.0    0.0    0.0 1 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area2" 22 "-" "-" 0 "-" "-" 0 "PUB" "Order" 13 "CustOrder" 40 1 2 3 20006 102181719   62.6    1.7   98.4    0.6 19973 19675
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area2" 22 "-" "-" 0 "-" "-" 0 "PUB" "Order" 13 "OrderDate" 41 1 1 2 370 428664   14.2    2.7 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area2" 22 "-" "-" 0 "-" "-" 0 "PUB" "Order" 13 "OrderNum" 39 1 1 2 145 943417   79.7    1.4    0.0    0.0 144 0
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area2" 22 "-" "-" 0 "-" "-" 0 "PUB" "Order" 13 "OrderStatus" 42 1 1 2 370 430144   14.2    2.7 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area2" 22 "-" "-" 0 "-" "-" 0 "PUB" "Order" 13 "SalesRep" 43 1 1 2 370 428294   14.2    2.7 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area3" 23 "-" "-" 0 "-" "-" 0 "PUB" "OrderLine" 14 "ItemNum" 45 1 1 1 1 3    0.0    1.0 "-" "-" "-" "-"
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area3" 23 "-" "-" 0 "-" "-" 0 "PUB" "OrderLine" 14 "OrderLine" 44 1 2 2 18 133998   91.2    1.2   94.0    6.0 17 16
/EdqpHgDw5awFMdPYAfWMg" "custdb1" "Index Area3" 23 "-" "-" 0 "-" "-" 0 "PUB" "OrderLine" 14 "OrderLineStatus" 46 1 1 1 1 3    0.0    1.0 "-" "-" "-" "-"
```


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Client Database-Request Statement Caching Configurable, Online-Changeable, and Enabled Where You Need It

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What & Why It Matters

- Client Statement Cache (CSC) lets a DBA monitor exactly what ABL or SQL database activity a client is running, in real time
- The DBA can track client activity and identify line numbers within an ABL program or SQL statement associated with database requests
- Invaluable for troubleshooting runaway queries, unexpected lock contention or performance problems tied to specific client code
- 13.0 focuses on making CSC easier to turn on, size and target without blanket overhead on every connection

The Problem

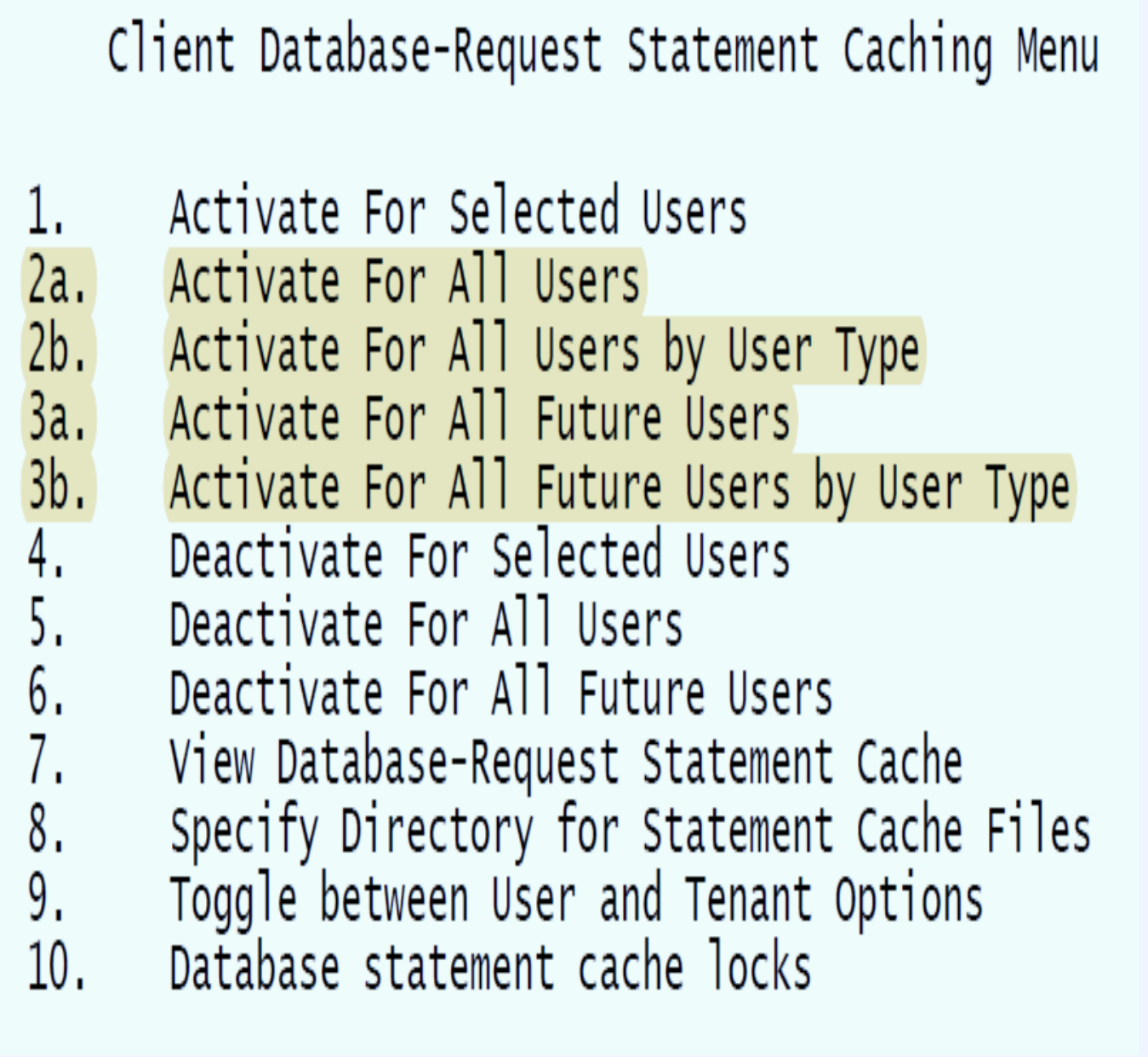
- CSC could not be enabled at database startup
- To minimize overhead of managing specific users, DBAs often enable CSC for all users and incur the associated resource and performance impact
- The overflow cache directory (for statements >256 bytes) could not be changed while the database was running
- The per-statement cache size was fixed at 256 bytes and the connection-reservation ceiling was fixed at ~12%—both too small for many real workloads

The Solution

- Enable client database-request statement caching as part of database startup
- Add database startup parameter that will configure ‘classes' of database connections to use statement caching based on the user connection type, for example local vs remote, ABL vs SQL, etc.
- Update the -clstmtcachedir parameter to allow online modification of the directory location

New & Enhanced Startup Parameters

Parameter	Range / Values	Default	Purpose
-clstmtCacheLevel	0 / 1 / 2	0 (none)	1 = single ABL line or SQL statement, 2 = full call stack
-clstmtCacheType	ALL, SELFSERVE, SQLREMC, ABLREMC, REMOTECLIENT	ALL	Limit caching to a specific class of client instead of every user
-clstmtCacheDir	Directory path	DB directory	Overflow location for cached statements >256 bytes—now online-changeable
-clstmtCacheSize	256 - 30000 bytes	256 bytes	Raise the per-statement shared-memory cache size for longer statements
-clstmtCacheAllocationFactor	0 - 100%	12%	% of connections reserved cache space, increase for larger connection pools



Enhancements to menu

Options 2 and 3 have been renamed to 2a and 3a

For backward compatibility, 2 and 3 will be supported as well

2b is a new option introduced to activate CSC for all users by user type

3b is a new option introduced to activate CSC for all future users by user type

Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Updating Parameters Online. No Restart Required.

- PROMON: R & D → 4 (Adjust Startup Parameters) → 16 (Client Statement Cache parameters)
- Query the _DbParams VST to see current, effective startup-parameter values at any time
- -clstmtCacheDir can be changed live via the PROMON Client Statement Cache menu option 8
- proutil dbname -C increaseto -clstmtCacheAllocationFactor n raises the connection-reservation ceiling on a live database

Diagnostic Data Collection Configurable Scope and Frequency for Critical Event Reporting

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What & Why It Matters

- The OpenEdge platform automatically captures diagnostic data when critical events occur, to speed up root-cause analysis
- Covers key event types: Lock Table Overflow, BI (Before-Image) Threshold crossings and System Errors
- This data is essential for support and engineering teams diagnosing production issues after the fact
- 13.0 gives DBAs control over how much diagnostic data is collected and how often, instead of an all-or-nothing model

The Problem

- Previously, diagnostic data was collected for ALL connections every single time a triggering event fired
- On a busy database with hundreds or thousands of connections, this could generate a flood of diagnostic output
- Repeated triggering events could overwhelm system resources and make the database temporarily unresponsive
- There was no way to scope collection to only the most relevant connections, or throttle how often the same event type was reported

New Startup Parameters

Parameter	Range	Default	Purpose
-diagTopN n	0 - 32000	0 (all connections)	Limit Lock Table Overflow and BI Threshold reporting to the top N most prominent offenders
-diagFrequency n	seconds, 0+	0 (disabled)	Throttle how often the same diagnostic event type (including System Errors) is re-reported

Updating Parameters Online. No Restart Required.

diagTopN and -diagFrequency parameters can be changed online using below options:

- Diagnostic Data Collection menu of PROMON (options R&D, 4, 14)
- Adjust Startup Parameters display of PROMON (options R&D, 4, 16)
- _DbParams VST

Key Takeaways

- PROBKUP: multithreaded backup now default for online & offline—shorter maintenance windows
- PROUTIL IDXCHECK: multi-threaded by default—faster health checks, faster recovery readiness
- PROUTIL IDXCOMPACT: -compactonly and true delete-placeholder cleanup keep indexes compact
- Client Statement Cache & Diagnostic Data Collection: online-changeable, configurable, throttled

News You Can Use

- Learn how to access the **OpenEdge AI Assistant**
- Get started with the **OpenEdge MCP Server**
- Book a demo of **Progress Agentic RAG**
- Modernize to **OpenEdge 12.8** today
- Explore what's new in **OpenEdge 13.0**
- Work with **Progress Professional Services** to accelerate adoption
- Learn about **security capabilities** across OpenEdge
- Explore the **developer tools** available for OpenEdge today
- Connect with us: **Discord, Community, CVP and more**



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.



Questions & Discussion

Soofia Shaik | Principal QA Engineer & Product Owner, OpenEdge Database

Thank You!