SQL Server Performance Audit

A mid-sized SQL Server estate (43 databases)

REDACTED SAMPLE. Produced from a real SQL Server instance. All server, database, schema and object names have been replaced with stable tokens. Every number shown is genuine.

Engagement	SAMPLE-001
Instance	SRV_01 — SQL Server 2019 Standard, 8 cores
Database examined	DB_02
Collected	2026-10-29
Report generated	2026-10-30 04:38 UTC
Diagnostic pack	v0.1.0
Prepared by	Ward Minson, QueryLens

1. Executive Summary

This audit examined **6 of 6** diagnostic areas and identified **9 findings**, of which **5** are high severity or above. **5** can be resolved with little or no downtime and should be taken first if your change window is short.

WHAT THIS MEANS

- **Corruption would go undetected: CHECKDB has never run** — 24 database(s) have no record of a successful DBCC CHECKDB. Page corruption on these databases would be discovered by a user reading a broken page, or by a restore that fails — not by you, and not in time. Backups of an already-corrupt database preserve the corruption.
- **Parallelism settings left at their defaults** — MAXDOP and/or cost threshold for parallelism are at values that no longer suit this hardware. These are the cheapest meaningful performance changes available on this instance.
- **Full backups are stale or absent** — 13 database(s) have no full backup in the last 7 days.
- **Integrity checks have lapsed** — 16 database(s) have not passed an integrity check in over 14 days; the worst is DB_27 at 4127 days.
- **Autogrowth settings will stall the application mid-transaction** — 28 database file(s) use percentage growth or a fixed increment small enough to cause frequent growth events, including 14 using percentage growth.

COVERAGE

Area	Rows examined
Wait statistics	50
Top query consumers	30
Index health	49
Backups and integrity	43
Configuration and tempdb	111
Blocking and deadlocks	1

2. Top 5 Prioritised Findings

Ranked by **severity × confidence × (6 − effort)**. The ranking is computed, not opinion — you can reproduce it from the table below.

#	Finding	Severity	Confidence	Effort	Impact	Objects
1	Corruption would go undetected: CHECKDB has never run	**Critical**	Certain	Low	100	24
2	Parallelism settings left at their defaults	**High**	Strong	Trivial	80	2
3	Full backups are stale or absent	**High**	Certain	Low	80	13
4	Integrity checks have lapsed	**High**	Certain	Low	80	16
5	Autogrowth settings will stall the application mid-transaction	**Moderate**	Certain	Trivial	75	28

1. Corruption would go undetected: CHECKDB has never run

Integrity · Severity **Critical** · Confidence **Certain** · Effort **Low** · Impact **100**

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

24 database(s) have no record of a successful DBCC CHECKDB. Page corruption on these databases would be discovered by a user reading a broken page, or by a restore that fails — not by you, and not in time. Backups of an already-corrupt database preserve the corruption.

SQL Server records the timestamp of the last successful integrity check on each database's boot page. These databases carry no such timestamp. Run CHECKDB against a restored copy on a non-production server to avoid the production I/O cost, then establish a weekly schedule.

EVIDENCE — 24 ROWS, FIRST 5 SHOWN

database_name	recovery_model_desc	last_known_good_checkdb
DB_02	FULL	—
DB_05	FULL	—
DB_06	FULL	—
DB_09	SIMPLE	—
DB_11	FULL	—

2. Parallelism settings left at their defaults

Configuration · Severity **High** · Confidence **Strong** · Effort **Trivial** · Impact **80**

MAXDOP and/or cost threshold for parallelism are at values that no longer suit this hardware. These are the cheapest meaningful performance changes available on this instance.

A cost threshold of 5 dates from 1997 hardware and sends trivial queries parallel, where coordination costs more than the work. MAXDOP at 0 lets a single query consume every core. Together they manifest as CXPACKET / CXCONSUMER waits that look like a hardware problem and are not.

EVIDENCE — 2 ROWS, FIRST 2 SHOWN

setting_name	observed_value	recommended_value	note
cost threshold for parallelism	5	50	The default of 5 dates from 1997 hardware and sends trivial queries parallel.
max degree of parallelism	6	8	Default 0 lets one query consume every core; on a wide box this manifests as CXPACKET / CXCONSUMER waits.

3. Full backups are stale or absent

Backup / RPO · Severity **High** · Confidence **Certain** · Effort **Low** · Impact **80**

13 database(s) have no full backup in the last 7 days.

Backup history here is read from msdb. A database with no recorded full backup either has never been backed up, or is backed up by a tool that does not register with SQL Server — which is itself worth confirming, because an unregistered backup is one nobody can restore under pressure.

EVIDENCE — 13 ROWS, FIRST 5 SHOWN

database_name	recovery_model_desc	last_full	full_age_hours
DB_07	FULL	2026-10-15 02:01:08	357
DB_08	FULL	2026-10-15 02:01:09	357
DB_09	SIMPLE	2026-10-15 02:01:18	357
DB_10	FULL	2026-10-15 02:01:19	357
DB_05	FULL	2026-10-15 02:01:20	357

4. Integrity checks have lapsed

Integrity · Severity **High** · Confidence **Certain** · Effort **Low** · Impact **80**

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

16 database(s) have not passed an integrity check in over 14 days; the worst is DB_27 at 4127 days.

The window between corruption occurring and corruption being detected is the window in which clean backups age out of retention. Weekly CHECKDB keeps that window shorter than your retention period.

EVIDENCE — 16 ROWS, FIRST 5 SHOWN

database_name	last_known_good_checkdb	checkdb_age_days
DB_27	2015-07-12 14:36:07.783000	4127
DB_07	2022-05-21 19:56:55.120000	1622
DB_08	2022-05-21 19:56:55.690000	1622
DB_25	2022-05-21 19:54:06.980000	1622
DB_31	2022-05-21 19:56:41.023000	1622

5. Autogrowth settings will stall the application mid-transaction

Configuration · Severity **Moderate** · Confidence **Certain** · Effort **Trivial** · Impact **75**

28 database file(s) use percentage growth or a fixed increment small enough to cause frequent growth events, including 14 using percentage growth.

Every growth event stalls the writes that triggered it. A 1 MB increment on a large, busy file means thousands of stalls; percentage growth compounds, so the events get slower exactly as the database gets bigger. Percentage growth on a log file also drives high VLF counts, which slow recovery, log backups and startup.

EVIDENCE — 28 ROWS, FIRST 5 SHOWN

setting_name	observed_value	recommended_value	note
file_growth:DB_03:FILE_33	1MB	256MB	Percentage growth compounds; very small fixed growth causes frequent stalls.
file_growth:DB_03:FILE_34	10%	64MB	Percentage growth compounds; very small fixed growth causes frequent stalls.
file_growth:DB_04:FILE_35	1MB	256MB	Percentage growth compounds; very small fixed growth causes frequent stalls.
file_growth:DB_04:FILE_36	10%	64MB	Percentage growth compounds; very small fixed growth causes frequent stalls.
file_growth:DB_07:FILE_61	1MB	256MB	Percentage growth compounds; very small fixed growth causes frequent stalls.

3. Full Findings Register

Integrity

Rule	Finding	Severity	Effort	Impact	Objects
INT-001	Corruption would go undetected: CHECKDB has never run	Critical	Low	100	24
INT-002	Integrity checks have lapsed	High	Low	80	16

Configuration

Rule	Finding	Severity	Effort	Impact	Objects
CFG-001	Parallelism settings left at their defaults	High	Trivial	80	2
CFG-003	Autogrowth settings will stall the application mid-transaction	Moderate	Trivial	75	28
CFG-002	tempdb is configured for allocation contention	High	Moderate	48	2
CFG-004	Instance settings left at defaults	Low	Trivial	40	3

Backup / RPO

Rule	Finding	Severity	Effort	Impact	Objects
BKP-002	Full backups are stale or absent	High	Low	80	13

Indexing

Rule	Finding	Severity	Effort	Impact	Objects
IDX-001	Duplicate indexes are charging you on every write	Moderate	Moderate	36	49

Performance

Rule	Finding	Severity	Effort	Impact	Objects
WAIT-001	Dominant wait types (insufficient history)	Informational	Moderate	3	10

4. Fix Scripts Appendix

Every script below is provided for **your** review and execution. Test in a non-production environment first. Scripts marked as requiring a maintenance window change behaviour in ways that are not instantly reversible.

INT-001 — Corruption would go undetected: CHECKDB has never run

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

```
-- Run against a RESTORED COPY on a non-production instance first.
-- PHYSICAL_ONLY is materially cheaper and catches the corruption
-- classes that matter most; run a full CHECKDB when you can afford it.
DBCC CHECKDB (N'<database>') WITH NO_INFOMSGS, PHYSICAL_ONLY;
```

ROLLBACK

```
-- Read-only operation. Nothing to roll back.
```

CFG-001 — Parallelism settings left at their defaults

```
EXECUTE sp_configure 'show advanced options', 1;
RECONFIGURE;
EXECUTE sp_configure 'cost threshold for parallelism', 50;
EXECUTE sp_configure 'max degree of parallelism', <recommended>;
RECONFIGURE;
```

ROLLBACK

```
-- Both settings take effect immediately and revert the same way.
EXECUTE sp_configure 'cost threshold for parallelism', <previous>;
EXECUTE sp_configure 'max degree of parallelism', <previous>;
RECONFIGURE;
```

BKP-002 — Full backups are stale or absent

```
BACKUP DATABASE [<database>]
  TO DISK = N'<path>\<database>_FULL.bak'
  WITH COMPRESSION, CHECKSUM, INIT, STATS = 10;
```

ROLLBACK

```
-- Additive. Taking a backup changes no database state.
```

INT-002 — Integrity checks have lapsed

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

```
DBCC CHECKDB (N'<database>') WITH NO_INFOMSGS, PHYSICAL_ONLY;
```

ROLLBACK

```
-- Read-only operation. Nothing to roll back.
```

CFG-003 — Autogrowth settings will stall the application mid-transaction

```
ALTER DATABASE [<database>]
  MODIFY FILE (NAME = N'<logical_name>', FILEGROWTH = 256MB);
-- Log files: 64MB is usually the better increment.
-- Pre-size files to their expected size so growth is rare.
```

ROLLBACK

```
ALTER DATABASE [<database>]
  MODIFY FILE (NAME = N'<logical_name>', FILEGROWTH = <previous>);
```

CFG-002 — tempdb is configured for allocation contention

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

```
-- Requires a service restart to take full effect.
ALTER DATABASE tempdb
  ADD FILE (NAME = N'tempdev<N>', FILENAME = N'<path>\tempdb<N>.ndf',
            SIZE = <equal_to_others>MB, FILEGROWTH = 64MB);
```

ROLLBACK

```
-- Files added to tempdb can be removed and the change is undone
-- at the next restart; tempdb is rebuilt on every service start.
```

CFG-004 — Instance settings left at defaults

```
EXECUTE sp_configure 'show advanced options', 1;
RECONFIGURE;
EXECUTE sp_configure '<setting>', <recommended>;
RECONFIGURE;
```

ROLLBACK

```
EXECUTE sp_configure '<setting>', <previous>;
RECONFIGURE;
```

IDX-001 — Duplicate indexes are charging you on every write

> **REQUIRES MAINTENANCE WINDOW — DO NOT RUN BLIND.**

```
-- Disable first and observe for a full business cycle.
-- Disabling is instantly reversible; dropping is not.
ALTER INDEX [<index_name>] ON [<schema>].[<table>] DISABLE;

-- Only after a clean observation period:
-- DROP INDEX [<index_name>] ON [<schema>].[<table>];
```

ROLLBACK

```
-- Re-enable requires a rebuild, which is an online operation on
-- Enterprise Edition and offline on Standard.
ALTER INDEX [<index_name>] ON [<schema>].[<table>] REBUILD;
```

5. 30 / 60 / 90-Day Plan

First 30 days — stop the bleeding

- **Corruption would go undetected: CHECKDB has never run** (INT-001)
- **Parallelism settings left at their defaults** (CFG-001) — no downtime required
- **Full backups are stale or absent** (BKP-002) — no downtime required
- **Integrity checks have lapsed** (INT-002)
- **tempdb is configured for allocation contention** (CFG-002)

Days 30–60 — remove the recurring cost

- **Autogrowth settings will stall the application mid-transaction** (CFG-003)
- **Duplicate indexes are charging you on every write** (IDX-001)

Days 60–90 — keep it from coming back

- **Instance settings left at defaults** (CFG-004)
- **Dominant wait types (insufficient history)** (WAIT-001)
- Establish a weekly integrity-check and backup-verification schedule.
- Re-run this audit and compare. The second report is where trends appear.

Prepared by Ward Minson, QueryLens · Diagnostic pack v0.1.0 · Document hash sha256:834b1fe7becc3f1f
Recommendations must be tested in a non-production environment before production use. The client owns execution.