SQL Server Migration Tool

Administrator Handbook

Migrating databases and server objects between SQL Server instances, inside one Active Directory domain and across domains. Includes the final step: database standardization and Always On availability group join.

Tool version	1.3 (SQL-Migration.ps1, GUI language en-US)
Document version	1.0, 2026-11-01
Platform	Windows PowerShell 5.1, dbatools, SQL Server 2016 to 2025
Author	Uwe Janke, powershelldba.de

Contents

1 Overview and scope
2 Same domain or different domain: choosing the procedure
3 How the tool works
4 Prerequisites and preparation
5 The user interface
6 Procedure A: migration inside one domain (Direct)
7 Procedure B: migration across domains (Detour)
8 Migration methods: Backup/Restore and Detach/Attach
9 Server objects in detail
10 The final step: standardization and Always On
11 Acceptance checklist
12 Logs, evidence and cleanup
13 Rollback
14 Troubleshooting
15 Quick checklists
A Appendix: configuration reference and log categories

1 Overview and scope

The SQL Server Migration Tool (`SQL-Migration.ps1`) is a WinForms GUI on top of dbatools. It moves **databases** and **server objects** (logins, linked servers, SQL Agent jobs, credentials, proxies) from a source SQL Server instance to a target instance. It is built for the situations that are awkward with the standard tools: servers that cannot see each other, servers in different Active Directory domains, and large databases that must be copied through a controlled path.

This handbook is written for database administrators who run migrations. It explains the two procedures the tool offers, which one to pick, what the tool does automatically, and where you have to act yourself.

1.1 What is new in version 1.3

- **Final step after the database migration.** After the databases and objects are in place, the tool runs the sqmSQLTool function `Invoke-sqmDatabaseStandardization` on the migrated databases and, if the target is the primary replica of an Always On availability group, joins the databases to that group with `Add-sqmDatabaseToAG`.
- **House standards applied automatically**: orphaned users repaired, users without login removed, compatibility level raised to the server level, `TARGET_RECOVERY_TIME` set to 60 seconds, database owner set to `sa`.
- **Separate source and target rows** with their own login (Windows or SQL login) and their own Connect button. The counterpart row checks reachability and login and decides whether the Direct procedure is possible.
- **Live progress**: backup and restore percentage with remaining minutes on both servers, and file copy progress with throughput.
- **Simplified selection**: only the Databases tab is a real selection. The object types are chosen with the Migrate checkboxes. The other tabs are views.
- **System logins are excluded** from the login transfer (sa, `##...##`, NT SERVICE, NT AUTHORITY, BUILTIN).
- **English user interface and log texts** (en-US).

1.2 What the tool does not do

- It does not migrate SSIS catalog contents, SSRS content, replication, Service Broker routes or server-level triggers. Use the dedicated sqmSQLTool functions or tools for these.
- It does not change application connection strings. Redirecting the applications is part of your cut-over plan.
- It does not delete anything on the source. Source databases stay online (Backup/Restore) or are re-attached (Detach/Attach with the re-attach option).

2 Same domain or different domain: choosing the procedure

This is the most important decision of a migration. The tool offers two procedures. They differ in **how the data travels** and, above all, in **what happens to Windows logins and database users**.

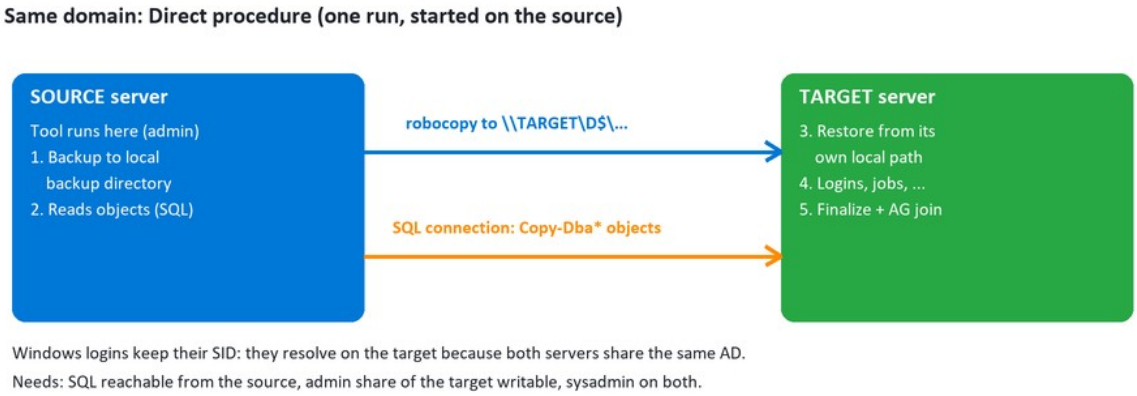


Figure 1: Direct procedure for source and target in the same domain.

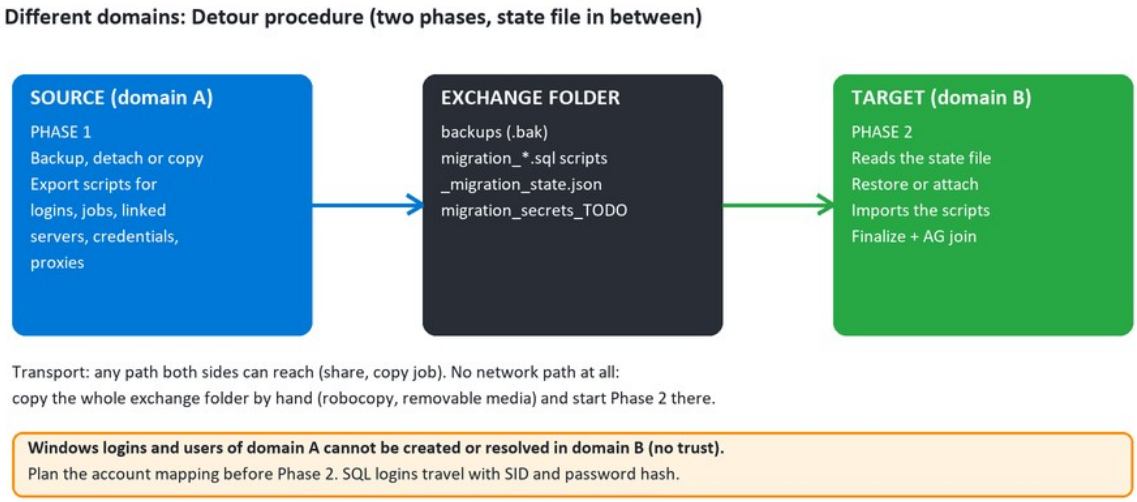


Figure 2: Detour procedure for source and target in different domains.

2.1 Side-by-side comparison

Topic	Same domain: Direct	Different domains: Detour
Where the tool runs	Once, on the **source** server. It sees both servers.	Twice: Phase 1 on the **source**, Phase 2 on the **target**. Each side sees only itself.
Runs in one pass	Yes. One click migrates databases and objects.	No. Two runs with a state file in between.
Data transport	Backup on the source, then `robocopy` into the target's **admin share** (`\\TARGET\D$\...`). Restore reads the target's own local path.	Backup, then `robocopy` to the **exchange folder**. Phase 2 copies from the exchange folder to the target and restores there.
Network requirements	SQL port (1433 by default) from source to target, SMB (445) to the target's admin share, same or trusted domain.	Only a path both sides can reach, or none: the exchange folder can be carried over by hand.

Topic	Same domain: Direct	Different domains: Detour
Windows logins and groups	Copied by `Copy-DbaLogin`. The accounts resolve through the shared AD and get the same SID on the target, so restored database users map instantly.	**Cannot be created** for accounts of the source domain. `CREATE LOGIN ... FROM WINDOWS` fails for an unknown domain. Those batches are skipped and logged.
SQL logins	Copied with SID and password hash.	Exported with SID and password hash in `migration_logins.sql`, imported batch by batch. Same result as Direct.
Database users of the old domain	Stay mapped (same SID).	**Become orphans.** If no login of the same name exists, the final step **drops them** (see 2.3).
Jobs, linked servers, credentials, proxies	Copied directly with `Copy-Dba*`.	Exported as scripts (`migration_jobs.sql`, ...) and imported in dependency order.
Orphaned AD login cleanup	Meaningful: finds logins whose accounts were deleted in AD.	Little to find, because foreign Windows logins were never created.
Always On join and standardization	Run at the end of the single pass.	Run at the end of Phase 2.
Typical use	Server replacement, version upgrade, move to new hardware inside one AD.	Company merger, move to a new forest, customer or hosting change, isolated networks.

2.2 How to decide

1. Are source and target in the **same AD domain** (or in domains with a working two-way trust)? Choose **Direct**.
2. Are they in **different domains without trust**, or is there **no network path** between them? Choose **Detour**.
3. Same domain, but the **admin share** of the target is blocked by policy or firewall? Choose **Detour** and use any share both sides can reach as the exchange folder.
4. Unsure? Leave **Procedure** on **Auto**. Press **Connect** on the counterpart row first: it tests TCP **and** the login and shows whether Direct is possible. When you start, Auto decides again by TCP reachability of the counterpart.

> **Warning** At start, Auto decides by TCP reachability of the counterpart only. It does **not** check whether the domains are the same or whether the login works. If the counterpart answers across two untrusted domains, Auto still chooses Direct. The copy into the admin share then needs Windows access to the target and fails in the pre-flight check before any backup starts. For a cross-domain migration set **Procedure = Detour** explicitly.

2.3 What a cross-domain migration does to Windows accounts

Windows logins are identified by a **SID**, not by name. A SID from domain A means nothing in domain B unless a trust exists. This has three consequences that you have to plan for:

1. **Windows logins and groups of domain A cannot be created on the target.** The import script contains `CREATE LOGIN [DOMA\user] FROM WINDOWS`. SQL Server rejects it with "Windows NT user or group not found". The tool logs the failed batch and continues. SQL logins are not affected.

2. **Database users based on those logins become orphans after the restore.** The restore brings the users along, but no login with the matching SID exists.
3. **The final step then removes them.** `Invoke-sqmDatabaseStandardization` first tries to re-map an orphaned user to a login **with the same name**. A user `DOMA\jsmith` finds no login `DOMA\jsmith` and, usually, no login named `DOMB\jsmith` either. Users without a login are dropped; schemas and roles they owned are handed over to `dbo` first. Permissions granted to the dropped users are lost.

Domain rule **Before a cross-domain migration**:

1. Script the database users and their permissions on the source (for example `Export-DbaUser`) and keep the files with the migration records.

2. Decide the account mapping (DOMA\jsmith becomes DOMB\jsmith) and create the target-domain logins **before Phase 2**, with the same names the users will be mapped by, or plan to re-create users and permissions from your scripts afterwards.

3. The GUI has no switch to skip the user removal. If you must keep orphaned users for manual mapping, run Phase 2 on a machine where sqmSQLTool is not available (the fallback cleanup keeps them, see 10.4) and run `Invoke-sqmDatabaseStandardization -SkipUserRemoval` later yourself.

Note Inside one domain none of this applies. `Copy-DbaLogin` keeps the SIDs, the restored users find their logins, and the standardization has nothing to drop.

3 How the tool works

3.1 Roles

At startup the tool asks which role this machine plays. You can also pass `-Role Source`, `-Role Target` or `-Role Auto`.

Role	Meaning
Source	Phase 1, or the single pass of the Direct procedure. Backup or detach, copy of the files, export of object scripts. Writes the state file in the Detour procedure.
Target	Phase 2 of the Detour procedure. Reads the state file, copies the files from the exchange folder, restores or attaches, imports the scripts, runs the final step.
Automatic	If a state file exists in the configured exchange path the tool starts as Target, otherwise as Source.

3.2 The exchange folder and the state file

The exchange folder holds everything that travels from source to target in the Detour procedure:

- the backup files (`.bak`) or the detached database files,
- `migration_logins.sql`, `migration_jobs.sql`, `migration_linkedservers.sql`, `migration_credentials.sql`, `migration_proxies.sql`,
- `_migration_state.json`: method, database list, object selection, file names, whether SQL logins are present, which scripts exist,
- `migration_secrets_TODO.txt`: credentials and linked servers whose secrets must be entered again on the target.

> **Note** Phase 2 looks for the state file in the path configured as `DefaultExchangePath` (default `\\exchange-server\SQLMigration\Backups`). If you carried the folder to another location, change `DefaultExchangePath` in the configuration file on the target before you start. Delete an old `_migration_state.json` before a new migration: a leftover file makes the Auto role pick Target.

3.3 Backup strategy: local first, then copy

The tool never asks the SQL Server service account to write to a network share. The backup goes first to the **default backup directory of the server** (where the service account always has write access). A `robocopy` run in the administrator context then copies it. In the Direct procedure the copy target is the admin share of the target server, and the restore reads the same file from the target's local path. The local backup stays on the source as a by-product and is never deleted automatically.

Before the (long) backup starts, the Direct procedure runs a **pre-flight check**: the target folder must be writable for the tool user and must have enough free space. With compression on, a shortage is only a warning. Without compression the run stops before the backup is taken.

3.4 Order of work

Step	Direct (one pass)	Detour Phase 1 (source)	Detour Phase 2 (target)
1	Backup or detach on the source	Backup or detach on the source	Read state file

Step	Direct (one pass)	Detour Phase 1 (source)	Detour Phase 2 (target)
2	robocopy into the target admin share	robocopy to the exchange folder, export object scripts, write state file	robocopy from exchange folder to the local target path
3	Restore or attach on the target	(end of Phase 1)	Restore or attach
4	Logins: Mixed Mode if needed, policy off, copy, policy on, AD cleanup		Logins: Mixed Mode if needed, policy off, import script, policy on, AD cleanup
5	Credentials, proxies, linked servers, agent jobs via `Copy-Dba*`		Credentials, proxies, linked servers, agent jobs from scripts
6	Final step: standardization, Always On join		Final step: standardization, Always On join

Logins and objects are processed **before** the final step on purpose: orphaned users can only be repaired after their logins exist.

4 Prerequisites and preparation

4.1 Requirements

Requirement	Detail
Operating system	Windows Server 2016 or later on source and target (Direct: the tool host is the source).
PowerShell	Windows PowerShell 5.1. The tool is started through `Start-SQLMigration.cmd`, which copies it to `C:\ProgramData\SQLMigration` and starts it elevated.
dbatools	Installed on every machine the tool runs on (`Install-Module dbatools`). For offline machines use a tested module package.
sqmSQLTool	Needed for the **final step** (standardization and Always On join). The tool looks for a loaded module, an installed module, or the source folder next to the tool. Without it, a simple fallback cleanup runs (see 10.4).
Rights	Local administrator on the machine running the tool. `sysadmin` on the instances it connects to.
SQL versions	Source SQL Server 2016 or later. Target 2016 to 2025, version equal to or newer than the source.
Certificates	With SQL Server 2022 and self-signed certificates keep **TrustServerCertificate** ticked.
Network, Direct	SQL port to the target and SMB (445) to its admin share (`\\TARGET\D$`).
Network, Detour	A share both sides can reach, or a manual transport of the exchange folder.
Disk space	Free space for one backup per database on the source (local backup directory), in the exchange folder, and on the target.

4.2 Pre-checks on the source

- Maintenance window agreed, applications informed or stopped.

- A current, **independent** full backup of every affected database exists (your safety net, separate from the migration backup).
- List of databases and object types fixed.
- Target: version at least the source version, enough disk space, data and log paths prepared, collation and compatibility requirements clarified.
- Exchange folder empty, no old `_migration_state.json`.
- A **WhatIf dry run** is planned.

4.3 Additional preparation for the same-domain procedure

- Run the tool as an account that is local admin on **both** servers and sysadmin on both instances.
- Check that `\\TARGET\D$` (or the drive that holds the target's default backup directory) is reachable from the source.
- If the target is part of an Always On availability group, check that it is the **primary** replica and that Automatic Seeding is enabled on all replicas (see chapter 10).

4.4 Additional preparation for the cross-domain procedure

- Agree how the exchange folder travels: a share reachable from both sides, a scheduled copy job, or manual transport. Check the free space on that medium.
- Export database users and permissions from the source and decide the account mapping (see 2.3).
- Create the target-domain logins that the mapped users will need, or plan the re-creation.
- Keep a list of the **SQL logins** and their purpose: the target may have to be switched to Mixed Mode, which restarts the SQL Server service.
- On the target, set `DefaultExchangePath` in the configuration if the folder is not at the default location.

5 The user interface

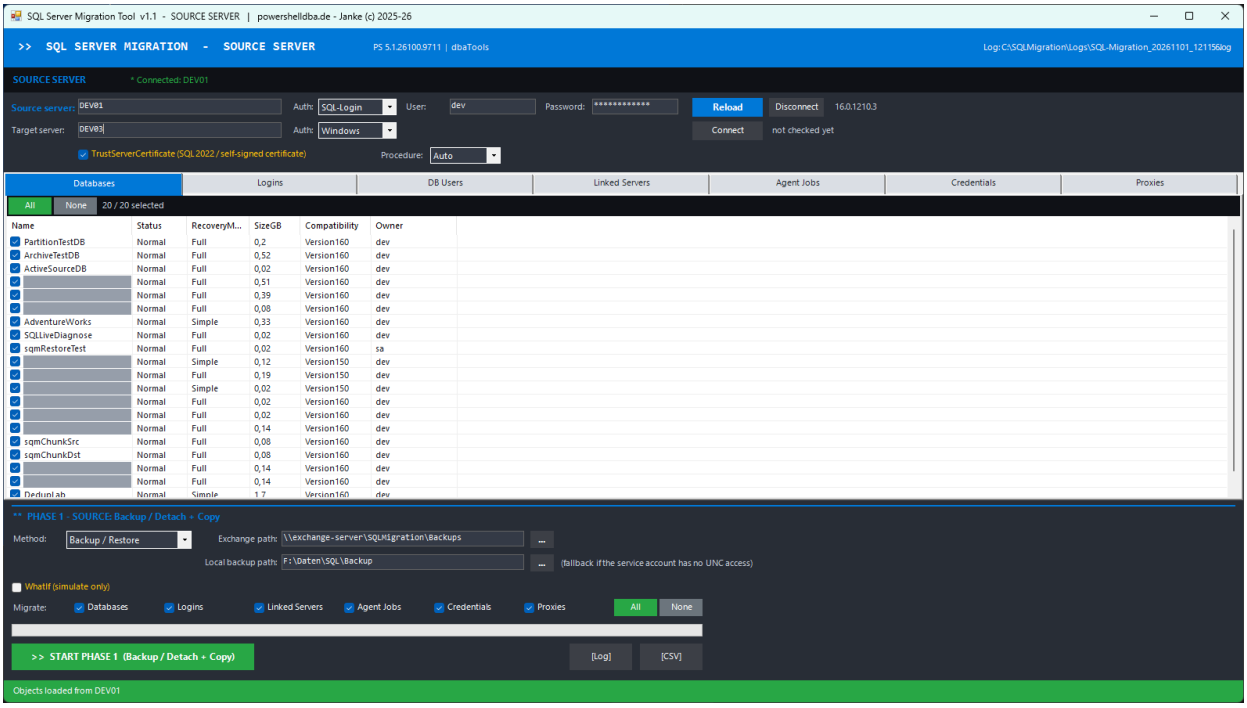


Figure 3: Source mode, connected to the source over a SQL login. The Databases tab shows the databases to migrate. Names of unrelated project databases are blacked out.

5.1 Connection area

Element	Function
Source server / Target server	Two rows, source always on top. The row of the role you started in is the active one (bold, role colour). It carries the buttons **Connect / Reload** and **Disconnect**. The other row is the counterpart.
Auth	Windows or SQL-Login, separately for each row. For SQL-Login, User and Password fields appear.
Connect (active row)	Connects and loads the objects. After loading, the button reads **Reload**.
Connect (counterpart row)	Checks that the other server answers on TCP and that the login works. The result text next to it says **DIRECT possible** or **DETOUR**. In Source mode with a successful check, the exchange path is set to the target's backup share automatically.
TrustServerCertificate	Applies to both servers. Keep it ticked for self-signed certificates.
Procedure	**Auto**, **Direct** or **Detour**. See chapter 2.

5.2 Tabs

Tab	Behavior
Databases	The only real selection. Tick the databases to migrate. **All** and **None** buttons, counter "n / m selected". Nothing is selected by you for the other tabs.

Tab	Behavior
Logins, DB Users, Linked Servers, Agent Jobs, Credentials, Proxies	View only. They show what exists on the connected server and how many objects will move. Whether an object type is migrated at all is decided by the **Migrate** row below. Every object of a ticked type is migrated.

5.3 Migration panel

Element	Function
Method	Backup / Restore (default) or Detach / Attach. Locked on the target (taken from the state file).
Exchange path	The transport folder. In the Direct procedure it points to the target's backup share. The path must pass an access test when you start.
Local backup path	Fallback if the server's own default backup directory cannot be determined.
WhatIf (simulate only)	Walks through the whole run and logs what it would do. Nothing is changed.
Re-attach source after detach	Only with Detach / Attach in Source mode. Attaches the source database again after the files were copied.
Migrate: Databases, Logins, Linked Servers, Agent Jobs, Credentials, Proxies, **All / None**	Selects the object types. Locked on the target (taken from the state file). Database users are not a separate type: they travel inside the databases.
Progress bar and text	Backup/restore percentage with remaining minutes on source and target, and copy progress (size, MB/s, minutes left). The text sits between the start button and the [Log] button.
START ... button	Reads **START PHASE 1** (Detour, source), **START MIGRATION** (Direct) or **START PHASE 2** (target). A summary dialog lists scenario, method, paths, databases, object types and the final step before anything starts.
[Log] and [CSV]	Open the current log file in Notepad and the CSV protocol.
Status bar	Shows the state and, during a run, the last log line. Colour: blue info, green success, amber warning, red error.

6 Procedure A: migration inside one domain (Direct)

Use this procedure when source and target belong to the same domain (or trusted domains) and the source can reach the target over SQL and SMB. Everything happens in one run, started on the **source**.

6.1 Step by step

1. Start `Start-SQLMigration.cmd` as administrator on the **source** server and choose the role **Source** (or pass `Source` as parameter).
2. The source row is pre-filled with the local machine. Check the name, choose **Auth**, press **Connect**. The objects load into the tabs.
3. In the **Target server** row enter the target instance, choose its **Auth**, press its **Connect**. The result must read **DIRECT possible**. The exchange path switches to the target's backup share.
4. Set **Procedure** to **Direct** (or leave **Auto** once the check says DIRECT).
5. On the **Databases** tab tick the databases to migrate.
6. In the **Migrate** row tick the object types. For a full server move tick everything.
7. Choose the **Method**. Backup / Restore keeps the source online and is the default.
8. Tick **WhatIf** and press **START MIGRATION** for a dry run. Read the log, untick WhatIf.
9. Press **START MIGRATION**. Read the summary dialog carefully, then confirm.
10. Watch the progress text. Wait for the completion dialog ("Migration completed").
11. Go through the acceptance checklist in chapter 11.

6.2 What happens during the run

1. **Per database**: pre-flight check of the target folder, backup into the source's default backup directory, `robocopy` into the target admin share, restore on the target from its local path (`WITH REPLACE`).
2. **Logins** (if ticked): Mixed Mode on the target if SQL logins exist and the target is Windows-only (the SQL service restarts), policy `New_Password_Policy` switched off, `Copy-DbaLogin` with SIDs, policy switched on again, orphaned AD login cleanup.
3. **Credentials, proxies, linked servers, agent jobs** (if ticked): `Copy-DbaCredential`, `Copy-DbaAgentProxy`, `Copy-DbaLinkedServer`, `Copy-DbaAgentJob`, in this order because of dependencies.
4. **Final step**: database standardization and, if applicable, Always On join (chapter 10).

> **Tip** A database that fails does not stop the run. The error is logged for that database and the next one starts. Check the log for `DIRECT-DB` errors before you declare the migration done.

7 Procedure B: migration across domains (Detour)

Use this procedure when source and target are in different domains without trust, or cannot see each other. It runs in two phases on two machines and exchanges data through the exchange folder.

> **Domain rule** Read section 2.3 first. Windows logins and users of the source domain do not survive the move. Prepare the account mapping **before** Phase 2.

7.1 Phase 1 on the source

1. Start `Start-SQLMigration.cmd` as administrator on the **source** and choose **Source**.
2. Connect the source row. Leave the counterpart row empty or check it: a failed check is fine and simply confirms **DETOUR**.
3. Set **Procedure** to **Detour**. The start button now reads **START PHASE 1**.
4. Tick the databases on the **Databases** tab and the object types in the **Migrate** row.
5. Choose the **Method** and enter the **Exchange path** (a share both sides can reach, or a local folder you will carry over).
6. Run a **WhatIf** pass, then the real run with **START PHASE 1**.
7. The tool writes backups or detached files, the `migration_*.sql` scripts, and the `_migration_state.json` file into the exchange folder.

> **Note** Passwords of SQL logins are exported as hashes together with the SIDs. The scripts therefore contain security-relevant data. Protect the exchange folder and delete it after the migration.

7.2 Transport

- **Shared path reachable from both sides:** nothing to do.
- **Copy job or manual transport:** copy the **whole** exchange folder (`.bak` files, all `migration_*` files and `_migration_state.json`) to a folder on the target side.
- On the target set `DefaultExchangePath` in `config\migration.config.json` to that folder, or place the files at the default path.

7.3 Phase 2 on the target

1. Start `Start-SQLMigration.cmd` as administrator on the **target**, choose **Target** (or **Automatic**, which finds the state file).
2. The tool reads the state file. **Method** and the **Migrate** selection are filled in and locked. The title reads **PHASE 2 - TARGET**.
3. Connect the target row.
4. Check the exchange path and the local backup path.
5. Press **START PHASE 2** and confirm the summary.
6. The tool copies the files from the exchange folder, restores or attaches, then processes logins and objects from the scripts, and ends with the final step.
7. Read the log for failed login batches (`LOGIN-IMPORT`). These are expected for Windows accounts of the source domain.

8. Apply your account mapping: create or fix target-domain logins, recreate users and permissions from your source scripts.
9. Work through the acceptance checklist in chapter 11 and delete the exchange folder.

7.4 Logins in Phase 2 in detail

Login type	Result in the target domain
SQL login	Created with the original SID and password hash. Mixed Mode is enabled if needed.
Windows login of the **source** domain	Not created (account unknown). Batch skipped, logged as warning.
Windows login of the **target** domain (same name exists)	Created normally if it was in the exported script.
Local Windows account of the source machine	Not created. Local accounts do not exist on the target.
System logins (sa, ##...##, NT SERVICE, NT AUTHORITY, BUILTIN)	Never exported or imported.

8 Migration methods: Backup/Restore and Detach/Attach

	Backup / Restore	Detach / Attach
Source availability	Source database stays **online**.	Source database is **taken offline** (detached).
Copy-only	Yes by default (`CopyOnlyBackup`), so the backup chain of the source is not disturbed.	Not applicable.
Compression and verify	On by default (`BackupCompression`, `VerifyBackup`).	Not applicable.
Data copied	One compressed `.bak` per database.	All data and log files at full size.
Rollback	Keep using the source. Nothing was changed there.	Use **Re-attach source after detach** or attach the files manually.
When to choose	Default for almost every migration.	Only when a backup is not possible or the files have to move as they are.

8.1 Direct procedure and the target path

For Backup/Restore the tool determines the target's default backup directory, converts it to the admin share path (`D:\MSSQL\Backup` becomes `\\TARGET\D$\MSSQL\Backup`), and restores from the local path. For Detach/Attach it uses the target's default data and log paths in the same way and builds a `CREATE DATABASE ... FOR ATTACH` statement.

8.2 WhatIf

With WhatIf the tool runs every decision, logs what it would do and changes nothing. Use it to verify paths, object counts and the scenario. WhatIf also covers the final step: the standardization reports what it would change and the Always On join reports its preconditions.

9 Server objects in detail

Object type	Direct (same domain)	Detour (different domains)	Notes
Databases	Backup, robocopy to admin share, restore	Backup to exchange folder, restore on target	Selected on the Databases tab.
Logins	`Copy-DbaLogin`, SIDs and hashes kept	`Export-DbaLogin` to `migration_logins.sql`, import per batch	System logins excluded. Windows logins of a foreign domain fail in Detour.
DB users	Travel inside the databases	Travel inside the databases	Orphans repaired or removed by the final step.
Linked servers	`Copy-DbaLinkedServer`	`migration_linkedservers.sql`	Stored passwords may not be readable: see secrets list.
Agent jobs	`Copy-DbaAgentJob`	`migration_jobs.sql`	Check job owners and proxies after the move.
Credentials	`Copy-DbaCredential`	`migration_credentials.sql`	Secrets may need to be entered again.

Object type	Direct (same domain)	Detour (different domains)	Notes
Proxies	`Copy-DbaAgentProxy`	`migration_proxies.sql`	Import order: credentials, proxies, linked servers, agent jobs.

9.1 Secrets list

Passwords of credentials and linked servers are sometimes not decryptable. For every such object the tool appends a line to `migration_secrets_TODO.txt` in the exchange folder. Work through the list on the target and enter the secrets again.

9.2 Mixed Mode and password policy

- If the source has **SQL logins** and the target accepts only Windows authentication, the tool switches the target to **Mixed Mode** and **restarts the SQL Server service**. The connection is rebuilt automatically. Connections to the target drop for a moment.
- Before logins are created the Policy-Based Management policy `New_Password_Policy` is switched **off** (if it exists), so that imported password hashes are not rejected. After the import it is switched **on** again.

9.3 Orphaned AD login cleanup

After the login transfer, and only when **Logins** is ticked, the tool removes logins whose Active Directory account was deleted. The rules are conservative:

- only Windows users and groups with a domain SID (`S-1-5-21-...`),
- never system logins and never members of the `sysadmin` role,
- a login is removed only when AD **positively** reports that the SID cannot be resolved. If the domain controller cannot be reached the login is kept and a warning is logged ("AD check skipped").

> **Warning** In a cross-domain migration run the target on a machine that can resolve the accounts you want to keep. A login whose SID the target cannot resolve is treated as deleted. Run the migration with **WhatIf** first and read the `AD-CLEAN` lines.

10 The final step: standardization and Always On

The final step runs at the very end of the Direct procedure and of Phase 2, whenever the **Databases** type was migrated. It needs the sqmSQLTool module. The start dialog shows the version it found, or warns that it is missing.

10.1 Database standardization

`Invoke-sqmDatabaseStandardization` brings every migrated database to the house standard. It returns one result row per action, and every row is written to the log (category `STANDARD`).

Step	What it does	Remarks
1 FixOrphanUser	Maps users whose SID matches no login to a login of the **same name** (`ALTER USER ... WITH LOGIN`).	Typical after a restore onto another server.
2 RemoveUserWithoutLogin	Drops the remaining users without login. Schemas and roles they own go to `dbo` first, in one transaction with the drop.	Never dropped: dbo, guest, system users, users created WITHOUT LOGIN, contained users, certificate and key users, and Windows users that reach the server through a Windows group login.
3 CompatibilityLevel	Raises the level to the level of the server version (160 on SQL Server 2022).	Never lowered. A new level can change execution plans: plan a performance check for critical applications.
4 TargetRecoveryTime	`TARGET_RECOVERY_TIME = 60 SECONDS` (indirect checkpoint).	
5 DatabaseOwner	Owner set to `sa`, found by SID `0x01`, so it also works when `sa` was renamed.	

Skipped completely: databases that are not ONLINE, read-only databases, secondary replicas of an availability group and database snapshots. System databases are never touched.

> **Domain rule** **Same domain:** the Windows logins arrived with their SIDs, so step 1 and 2 find (almost) nothing to do.
>
> **Different domains:** steps 1 and 2 are where the old-domain users disappear. See 2.3 before you run Phase 2.

10.2 Always On availability group join

If the target instance has Always On enabled and is the **primary replica of exactly one** availability group, the tool calls `Add-sqmDatabaseToAG` for the migrated databases. The function:

- sets the recovery model to **Full** if necessary and takes a **full backup** right after, so the log chain exists (the same applies to a Full database that never had a full backup),
- checks TDE-encrypted databases and their preconditions before any change,
- **drops** an existing database of the same name on the secondary replicas,
- adds the database with **Automatic Seeding**, one database after the other.

Situation on the target	What the tool does
Always On not enabled, or target is not a primary replica	Logs "no AG join" and finishes.
Primary of exactly one AG	Joins the databases to that AG.
Primary of several AGs	Does **not** guess. Logs the AG names and a hint to run `Add-sqmDatabaseToAG -AvailabilityGroup <Name> -Database ...` yourself.
Status cannot be determined	Logs a warning and skips the join.
sqmSQLTool missing	Fallback cleanup (see 10.4). The AG join is skipped with a warning: do it manually.

Warning Plan the AG join like a production change:
- Automatic Seeding must be enabled on **all** replicas (`Invoke-sqmSqlAlwaysOnAutoseeding` can do this).
- A database with the same name on a secondary is **dropped** before seeding.
- The seeding traffic and the full backup add load. Run large migrations in a window.
- TDE databases need SQL Server 2019 or later on every replica and the certificate on every secondary (`-SyncTdeCertificate` of `Add-sqmDatabaseToAG`).

A **WhatIf** run reports per database whether the preconditions are met.

10.3 Direct procedure and the AG

In the Direct procedure the restore happens on the target instance first. The databases are joined to the AG at the end, after the logins and objects are in place. The same order applies to Phase 2 of the Detour procedure.

10.4 Without sqmSQLTool

If sqmSQLTool is not available the tool falls back to `Invoke-PostRestoreCleanup`: orphaned users are repaired with `Repair-DbaDbOrphanUser` and the owner is set to `sa`. This fallback **does not remove** orphaned users, does not raise the compatibility level, does not set `TARGET_RECOVERY_TIME` and does **not join the AG**. A warning in the start dialog, the status bar and the log says so.

11 Acceptance checklist

Go through this list on the target before you hand the system over.

Check	How	Both procedures
All expected databases present and ONLINE	`SELECT name, state_desc FROM sys.databases`	Yes
`DBCC CHECKDB` clean	Run on every migrated database	Yes
Recovery model, compatibility level, `TARGET_RECOVERY_TIME`	Compare with the standard (level of the server, 60 s)	Yes
Owner is `sa`	`SELECT name, SUSER_SNAME(owner_sid) FROM sys.databases`	Yes
Logins present with correct SIDs	Compare source and target	Yes
Windows accounts mapped	Create target-domain logins, re-create users and permissions	**Detour only**
No unexpected orphaned users	Run an orphan check per database	Yes
Mixed Mode if SQL logins exist	Server properties, security	Yes
`New_Password_Policy` active again	Policy-Based Management	Yes
Jobs, linked servers, credentials, proxies run	Run key jobs, test linked servers	Yes
Secrets entered	Work through `migration_secrets_TODO.txt`	Yes
AG join result	Dashboard, `sys.dm_hadr_database_replica_states`, log category `AG`	If applicable
Connection strings redirected, application tests passed	Application owners	Yes
Source databases retired only after acceptance	Take offline first, drop later	Yes

12 Logs, evidence and cleanup

- **Log file**: `C:\SQLMigration\Logs\SQL-Migration_<date>_<time>.log` (button **[Log]**). Every step with timestamp, level, category and detail.
- **CSV protocol** (button **[CSV]**): the same events for filtering in a spreadsheet.
- **Error dialog**: shows the complete exception chain and can be copied.
- **Local backup files** are not deleted automatically. Remove them after acceptance.
- **Exchange folder**: delete it after the migration. It contains backups and, in the Detour procedure, password hashes in the login script.
- `_migration_state.json` must be removed before the next migration, otherwise the Auto role picks Target.

12.1 Reading the log

Levels: `INFO`, `STEP` (a phase starts), `SUCCESS`, `WARN` (continue, but look), `ERROR` (this item failed). Categories name the part of the tool; the most useful ones are listed in the appendix.

13 Rollback

Situation	Rollback
Before restore or attach on the target	Nothing has changed. Cancel.
Backup / Restore method	The source stayed online and unchanged. Keep the applications on the source. Remove the restored databases on the target.
Detach / Attach method	The source was detached. Use **Re-attach source after detach** or attach the original files manually.
Logins created on the target	Remove them if the target was a fresh server. On an existing server compare with the pre-migration state.
AG join already done	Remove the database from the AG (`Remove-DbaAgDatabase`), then drop it on the replicas.
Anything unclear	Restore the independent full backup you took before the migration.

14 Troubleshooting

Symptom	Cause and action
"dbaTools is not installed"	Run `Install-Module dbatools`.
Certificate or connection error	Tick **TrustServerCertificate**. Check server name, port and firewall.
"Exchange path not reachable"	Check the UNC path and the rights of the **running admin**. In Direct the path is the target's admin share.
Counterpart check says reachable but login failed	Wrong credentials, or no trust between the domains. For a cross-domain move use **Detour**.

Symptom	Cause and action
Direct stops for a database with an error about the target path ("not writable", access denied)	The admin share of the target is not reachable or not writable for the tool user (policy, firewall, no trust). Use **Detour** or fix the share rights.
"Not enough space on target"	Free space on the target backup drive, or enable backup compression.
Backup is created locally	Intended: backup goes to the local backup directory first, then `robocopy` copies it.
robocopy error (exit code 8 or higher)	Check paths, the admin's rights on both ends and free space.
Phase 2 finds no state file	Phase 1 did not finish, or `DefaultExchangePath` on the target points elsewhere.
Windows logins missing after Detour	Expected for accounts of the source domain (see 2.3). Check `LOGIN-IMPORT` warnings, create target-domain logins.
Database users missing after Phase 2	The final step dropped users without login (step 2 of the standardization). Recreate them from the user scripts exported from the source.
Users orphaned after restore (same domain)	Should not happen. Check that the Windows logins were copied (`LOGIN` log lines) and rerun `Invoke-sqmDatabaseStandardization`.
"AD check skipped (inconclusive)"	The domain controller could not be reached. The login was kept. Safe, no action.
No AG join happened	Log category `AG`: target not primary, several AGs, or sqmSQLTool missing. Use `Add-sqmDatabaseToAG` manually.
AG join failed	Read the detail: Automatic Seeding disabled on a replica, TDE certificate missing, recovery model or backup problem.
Warning "sqmSQLTool is not installed"	Install the module or place it next to the tool. Until then only the fallback cleanup runs.
Jobs, linked servers, credentials or proxies missing after Detour	Check the `migration_*.sql` files and the `*-EXPORT` (Phase 1) and `*-IMPORT` (Phase 2) log categories. Enter secrets from the TODO list.
SQL service restarted unexpectedly	Mixed Mode was enabled because SQL logins exist (see 9.2).

15 Quick checklists

15.1 Same domain (Direct)

- [ ] Independent full backup, window agreed, WhatIf planned
- [ ] Tool on the **source**, role Source, source connected
- [ ] Target row connected, result **DIRECT possible**
- [ ] Databases ticked, object types ticked, method chosen
- [ ] WhatIf run read, then **START MIGRATION**
- [ ] Log checked for `DIRECT-DB`, `LOGIN`, `STANDARD` and `AG` problems
- [ ] Acceptance checklist, connection strings, application test
- [ ] Cleanup: local backups on source and target, old state file

15.2 Different domains (Detour)

- [ ] Independent full backup, window agreed, account mapping decided
- [ ] User and permission scripts exported from the source
- [ ] Procedure set to **Detour**, Phase 1 run on the source
- [ ] Exchange folder complete (`.bak`, `migration_*`, `_migration_state.json`) and transported
- [ ] `DefaultExchangePath` on the target points to the folder
- [ ] Phase 2 run on the target, `LOGIN-IMPORT` warnings reviewed
- [ ] Target-domain logins created, users and permissions re-created
- [ ] Secrets entered, final step and AG join result checked
- [ ] Acceptance checklist, exchange folder deleted

Appendix A Configuration reference and log categories

A.1 Configuration file

The tool reads `config\migration.config.json` next to the script. If the file is missing, the defaults below apply. You can also pass `-ConfigFile <path>`.

Key	Default	Meaning
`DefaultExchangePath`	`\\exchange-server\SQLMigration\Backups`	Exchange folder. Phase 2 looks for the state file here.
`DefaultLocalBackupPath`	`F:\Daten\SQL\Backup`	Fallback for the local backup directory.
`DefaultLogPath`	`C:\SQLMigration\Logs`	Log and CSV directory.
`LogFilePrefix`	`SQL-Migration`	Log file name prefix.
`ConnectionTimeout`	`30`	Seconds.
`BackupCompression`	`true`	Compressed backups.
`VerifyBackup`	`true`	Verify after backup.
`CopyOnlyBackup`	`true`	Copy-only backups.
`DefaultMigrationMethod`	`BackupRestore`	Or `DetachAttach`.
`UncAccessTestTimeoutSec`	`10`	Timeout of the exchange path access test.
`StateFileName`	`_migration_state.json`	State file name.
`TrustServerCertificate`	`true`	Pre-ticks the checkbox.

A.2 Log categories

Category	Where it comes from
`BACKUP`, `RESTORE`, `DETACH`, `ATTACH`, `REATTACH`	Database steps (Detour phases).
`DIRECT`, `DIRECT-DB`	Direct procedure: single pass and per-database transfer.
`COPY`, `PATH`	robocopy and path determination.
`LOGIN`, `LOGIN-EXPORT`, `LOGIN-IMPORT`, `LOGIN-CHECK`	Login transfer (copy, export, import, SQL login detection).
`AUTHMODE`, `PBM`	Mixed Mode switch and password policy.
`AD-CLEAN`	Orphaned AD login cleanup.
`LINKED-SRV`, `AGENT-JOB`, `CREDENTIAL`, `PROXY`	Object copy in the Direct procedure.
`LS-EXPORT`, `JOB-EXPORT`, `CRED-EXPORT`, `PROXY-EXPORT` and `*-IMPORT`	Script export and import in the Detour procedure.
`SECRETS`	Entries for the secrets list.
`FINALIZE`, `STANDARD`, `AG`	Final step, standardization, Always On join.
`ORPHAN`, `OWNER`	Fallback cleanup without sqmSQLTool.

A.3 Terms

Term	Meaning
SID	Security identifier. Identifies a login or a Windows account independent of its name. A restored database user is linked to its login by SID.
Orphaned user	A database user whose SID matches no server login.
Exchange folder	The transport folder with backups, scripts and the state file.
State file	`_migration_state.json`, the hand-over from Phase 1 to Phase 2.
Direct / Detour	The two procedures. Detour is the two-phase path (internal name TwoPhase).
Automatic Seeding	Always On feature that streams the database to the secondary replicas without a manual restore.