Post Clone Scripts

GRIP ON SQL

2024-04-16

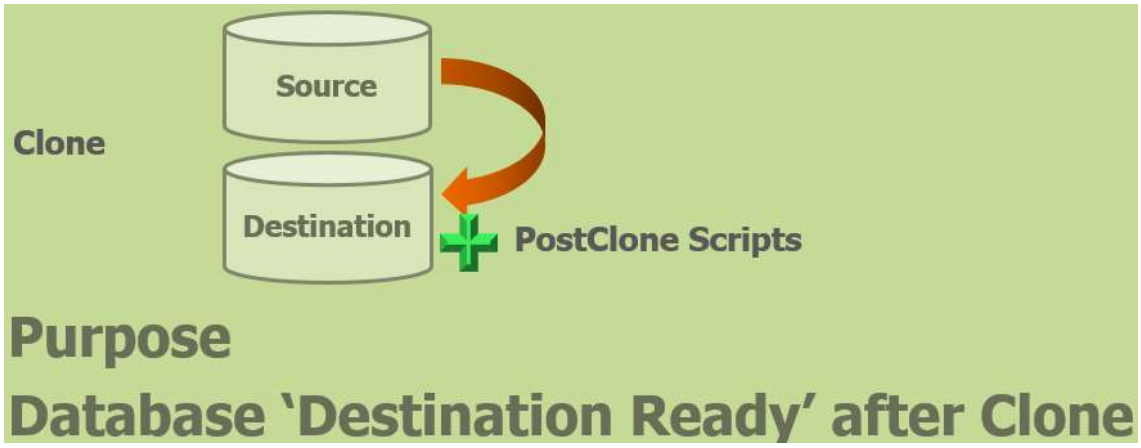
Contents

1 Post Clone Scripts 3
1.1 Post Clone Script - Types 3
1.2 Script Advice 4
2 Post Clone Script - Execute 5
3 Post Clone Script – Edit 7
3.1 Insert -> Statement 7
3.2 Insert -> RemoteProc 8
3.3 Insert -> Substitution 8
3.4 Generate -> Example 8
3.5 Generate -> Template 8
3.6 Generate -> PostClone 8
4 Post Clone Script – Example 9
5 Post Clone Script - Test 10
5.1 My Databases 10
5.2 Test Script 10
6 QGrip-Remote-Tmp stored procedures 12
6.1.1 QGrip-Remote-Tmp-Usp-Role-* 12
6.1.2 QGrip-Remote-Tmp-Usp-DBUser-Create 13
6.1.3 QGrip-Remote-Tmp-Usp-DBUser-Drop(All) 14
7 Problems dropping Database Users 15
7.1 Work around: EXECUTE AS 15
8 Password Safe update 17

1 Post Clone Scripts

Recommended documentation

Doc-Tab	Title
Basics	Database Alias
Basics	Substitution & Naming Convention
Jobs	Restore-Clone-Import Database



The purpose of the PostClone scripts is to adjust a database to fit in the destination environment after a Clone. Mostly, it includes dropping the Source environment database users and create the users needed in the Destination environment. Also common is changing content of Parameter and Configuration tables that is environment dependant.

A PostClone Script is a "normal" T-SQL script with SQL statements. To minimise the length and complexity of the script, Variables (QGrip Substitutions) and 'QGrip-Remote-Tmp' stored procedures should be used.

QGrip will prevent context switches to another database in the Post Clone scripts. The scripts should *only* run on the database that has been cloned.

1.1 Post Clone Script - Types

1. General
   - General Settings
   - Whole organisation
2. Updates
   - Config tables & Anonymise
   - Per Application
3. Users
   - Database Users
   - Per Application

There are 3 different Script types and they will always run in the same order.

Order	Script Type	Description
1	General	Can only be edited by QGrip Admin. It should only contain changes that apply to all databases in all environment and for all Applications.
2	Updates	Updates of Parameter and Configuration tables, anonymising data or other adjustments that are not users related.
3	Users	Drop the existing users and creating the Destination users.

Updates and Users scripts are created for a specific Application.

1.2 Script Advice

- Whole Organisation
  - One General Settings
- Per Application
  - One Other Updates
  - One Database Users

No choice is the best choice. We advise you to have *one* General Settings script and per application *one* Updates and *one* Users script. The script can even be empty or only contain a comment block.

Another advice is to always have the Post Clone scripts ready in case you need to quickly clone a database to another environment to solve errors in the production environment.

2 Post Clone Script - Execute

1. Create 'QGrip-Remote-Tmp'
2. Concatenate scripts
- **General Settings**
- **Other Updates**
- **Database Users**

3. Apply Substitutions (Find->Replace)
4. Run Script
5. Drop 'QGrip-Remote-Tmp'

During the PostClone stage of a clone (which is directly after the restore), the following is done:

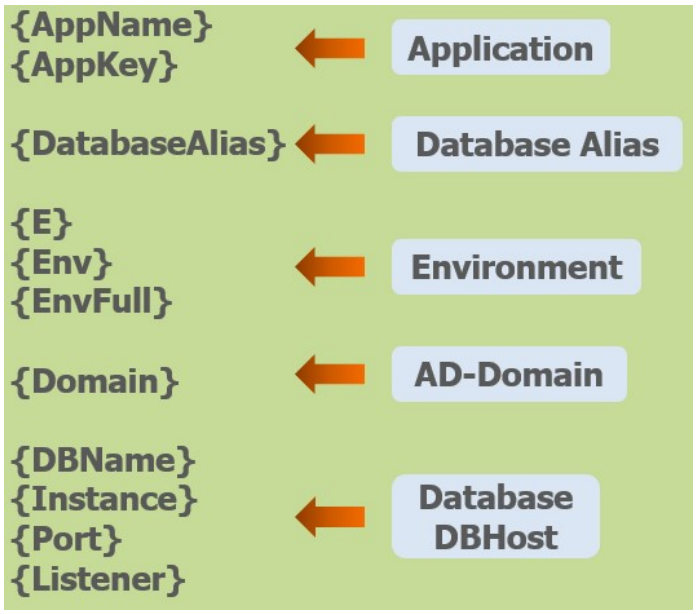
- Create 'QGrip-Remote-Tmp' stored procedures.
- Concatenate to one script
  - General script.
  - Updates script.
  - Users script.
- Substitution (Find/Replace) is performed on the script.
- Drop 'QGrip-Remote-Tmp' stored procedures.

Additional actions, after execute of the script:

- QGripDBHistory table is created/updated in the Database.
- Password Safe is updated for created SQL Logins.

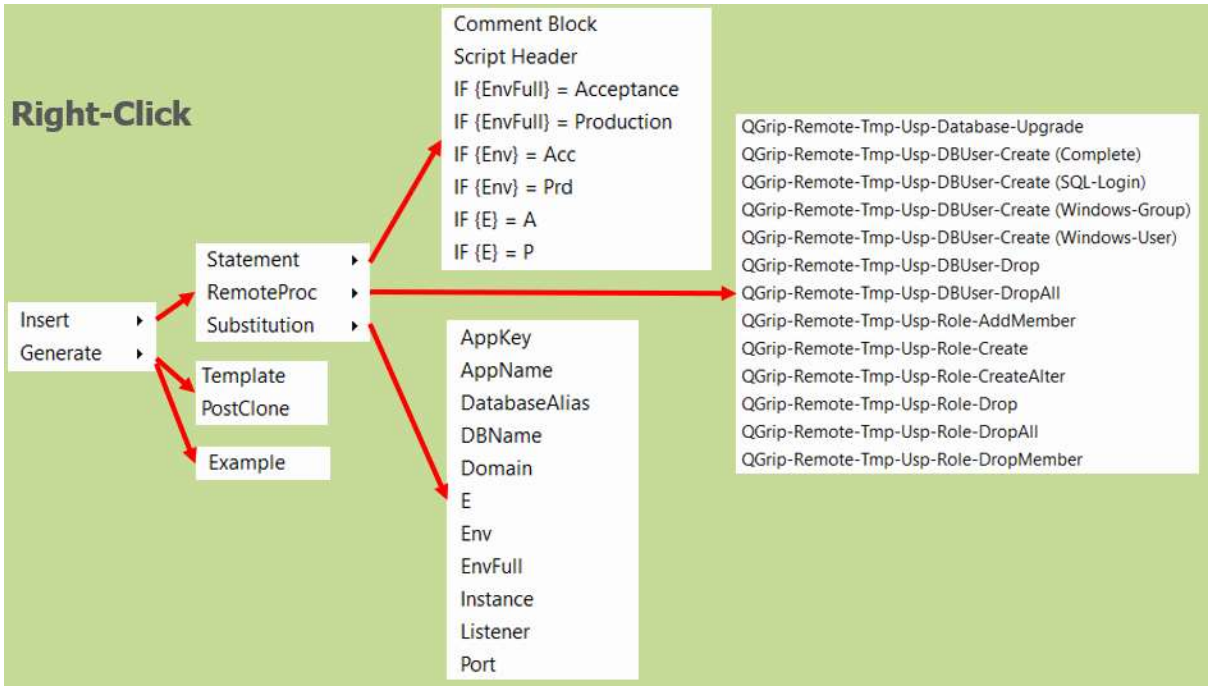
QGrip-Remote-Tmp-Usp-Database-Upgrade
QGrip-Remote-Tmp-Usp-DBUser-Create (Complete)
QGrip-Remote-Tmp-Usp-DBUser-Create (SQL-Login)
QGrip-Remote-Tmp-Usp-DBUser-Create (Windows-Group)
QGrip-Remote-Tmp-Usp-DBUser-Create (Windows-User)
QGrip-Remote-Tmp-Usp-DBUser-Drop
QGrip-Remote-Tmp-Usp-DBUser-DropAll
QGrip-Remote-Tmp-Usp-Role-AddMember
QGrip-Remote-Tmp-Usp-Role-Create
QGrip-Remote-Tmp-Usp-Role-CreateAlter
QGrip-Remote-Tmp-Usp-Role-Drop
QGrip-Remote-Tmp-Usp-Role-DropAll
QGrip-Remote-Tmp-Usp-Role-DropMember

The available 'QGrip-Remote-Tmp' stored procedures will be described in detail in a separate section below.

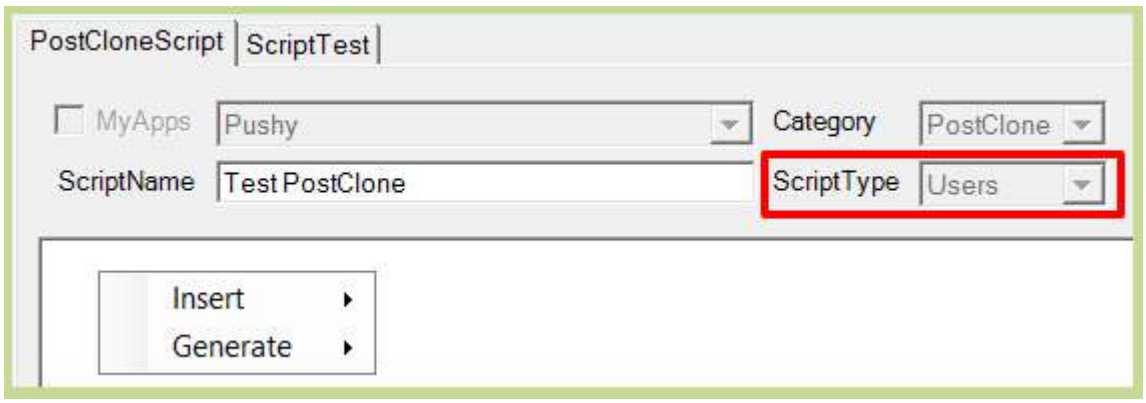


These are the Variables that can be used in the post clone scripts with their source.

3 Post Clone Script – Edit



When editing the post clone scripts in the QGrip-UI, a right-click in the script will open a menu to insert pieces of code or substitutions in the script. It is also possible to generate a Template, PostClone or Example.



The options you have in the menu depend on the current Script Type.
Inserts will insert a statement at the cursor position in the Script. The statement block will be inserted with the same amount of space from the beginning of the line.
Generate, will replace the existing script but only after you have accepted the “Overwrite” warning.

3.1 Insert -> Statement

Comment Block
Inserts an empty comment block

Script Header
Insert an empty script header that you will need to complete.

IF ('{EnvFull}' = 'Acceptance'), IF ('{Env}' = 'Acc'),IF ('{E}' = 'A')
Insert an if statement with Full, Short or Char Environment indication.

3.2 Insert -> RemoteProc

Inserts an EXEC QGrip-Remote-Tmp procedure call.

3.3 Insert -> Substitution

Inserts the Substitution variables (the Finds).

3.4 Generate -> Example

Generates an Example that will hopefully give you an idea of how to proceed. It does not use any application data.

3.5 Generate -> Template

Generates a Template based on the selected Application and Template information available at that moment. A really good starting point, but you need to check the whole script and remove stuff that is not applicable. In comparison with the PostClone described in the next section, the Template is much more sustainable, but it might need updating after changes to Databases, Logins and Database users.

3.6 Generate -> PostClone

Generates a PostClone script based on the selected Application and the current situation on the Instances / Databases. This is quick but it is a snapshot of the current situation. If a database is moved or users are changed, it will not work as originally intended.

4 Post Clone Script – Example

```
/*************************************************
* Upgrade Database to Instance Version (if needed)
*************************************************/
EXEC [dbo].[QGrip-Remote-Tmp-Usp-Database-Upgrade]
GO
```

General Settings: Upgrade

```
/*************************************************
* Update Database Config table
*************************************************/
IF ( '{DatabaseAlias}' IN ('Core') )
BEGIN
   UPDATE [dbo].[Database-Config-Table]
   SET [Environment]  = '{EnvFull}'
   ,   [InstanceName] = '{Instance}'
   ,   [PortNumber]   = '{Port}'
   ,   [DatabaseName] = '{DBName}'
END
GO
/*************************************************
* Anonymise
*************************************************/
IF ( '{EnvFull}' <> 'Production' )
BEGIN
   UPDATE [dbo].[Personal Data]
   SET [SocialSecurityNumber]  = 'xxxxxx'
END
GO
```

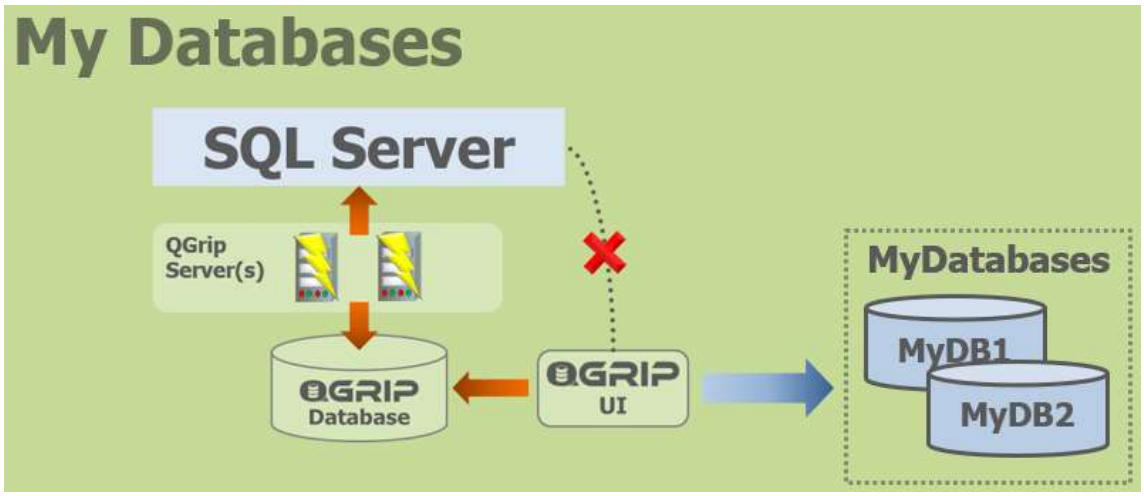
Other Updates

```
/*************************************************
* Drop all database users
*************************************************/
EXEC [dbo].[QGrip-Remote-Tmp-Usp-DBUser-DropAll]
GO
/*************************************************
* Create Logins / Database users
*************************************************/
IF ( '{DatabaseAlias}' IN ('Core') )
BEGIN
   EXEC [dbo].[QGrip-Remote-Tmp-Usp-DBUser-Create]
     @Login         = 'SQL_PA_{E}_Pushy-App_SvcUser'
   , @Integrated    = 'N'
   , @DefaultSchema = 'dbo'
   , @DefaultDB     = 'tempdb'
   , @DBRoles       = 'RoleSvcUser'
END
GO
```

Database Users

5 Post Clone Script - Test

5.1 My Databases

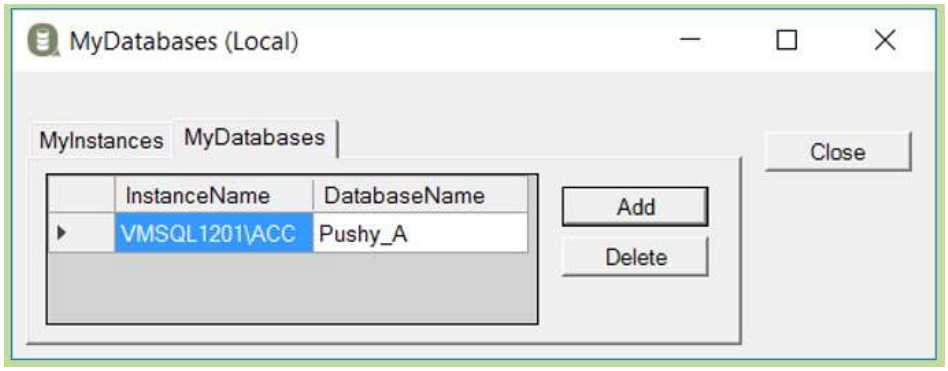


Normally, the QGrip-UI only communicates with the QGrip database and not with the SQL Server instance on the Infra. For testing Post Clone Scripts, there is an extra option, My Databases.

Preferably, the My Databases should be (local) databases on a separate location and definitely not Production databases.

QGrip UI: File -> MyDatabases (Local)

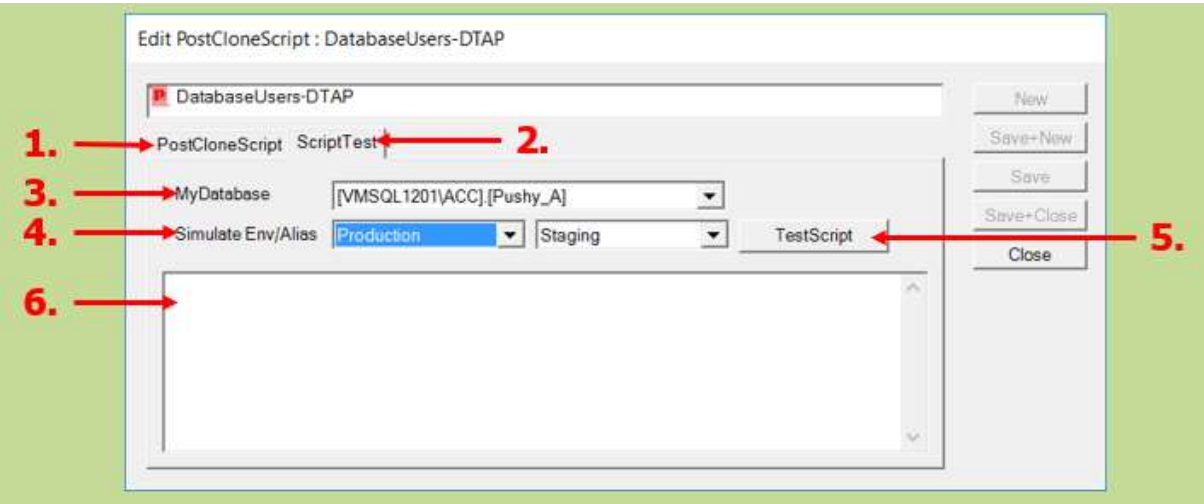
- **Add Instance**
- **Add Database**



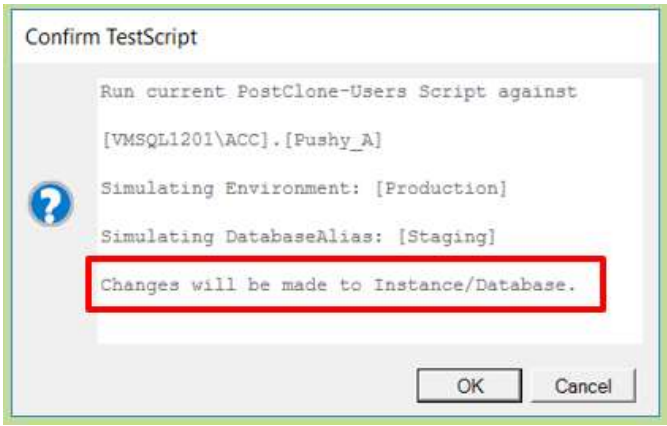
The My Databases can be added from the file menu. Start by adding the Instance and then the database. The current AD user is used to connect with Windows Authentication. The user needs to have sufficient right on the Instance/Database, to perform the actions in the Post Clone script.

Warning
The actions in the Post Clone script will be applied and changes will be made to Instance/Database!

5.2 Test Script



1. The script in the 'Script' tab will be executed, even if changes have not yet been saved.
2. Select the 'ScriptTest' tab.
3. Choose MyDatabase to execute against.
4. Choose Environment and Database Alias to simulate. QGrip will use these parameters in the Find/Replace.
5. Press the 'TestScript' button and confirm the action.
6. This is where the output of the script can be found.




```
Script Output:
INFO: Dropped DatabaseUser: [SQLLogin_3010_T_Zeus_Application]
INFO: Dropped DatabaseRole: [RoleFATester]
INFO: Dropped DatabaseRole: [RoleApplication]
INFO: Created DatabaseRole: [RoleFATester]
INFO: DatabaseRole: [RoleFATester] DBPermissions granted: [Execute]
INFO: Added [RoleFATester] as member of DatabaseRole [db_datareader]
INFO: Added [RoleFATester] as member of DatabaseRole [db_datawriter]
INFO: Created DatabaseRole: [RoleApplication]
INFO: DatabaseRole: [RoleApplication] DBPermissions granted: [Select,Insert,Delete,Update,Execute,References,Alter]
INFO: Created DatabaseUser: [SQLLogin_3010_T_Zeus_Application] from Login [SQLLogin_3010_T_Zeus_Application]
INFO: DefaultSchema: [dbo] set for DatabaseUser [SQLLogin_3010_T_Zeus_Application]
INFO: Added [SQLLogin_3010_T_Zeus_Application] as member of DatabaseRole [RoleApplication]
```

Example Output

6 QGrip-Remote-Tmp stored procedures

The QGrip-Remote-Tmp procedures is a set of stored procedures that will be available during the PostClone stage and are meant to make it easier to manipulate Database Roles and User (Logins). QGrip uses the same set of stored procedures when creating AppObjects. The QGrip-Remote-Tmp statements can be pasted from the Script Menu.

6.1.1 QGrip-Remote-Tmp-Usp-Role-*

QGrip-Remote-Tmp-Usp-Role-Create

Parameter	Type	Mandatory	Description
@Role	String	Yes	'<Role>', the name of the Role

Creates the database role <Role> if it does not already exist. No permissions are granted to the role.

QGrip-Remote-Tmp-Usp-Role-CreateAlter

Parameter	Type	Mandatory	Description
@Role	String	Yes	'<Role>', the name of the Role
@DBPermissions	String	No	'<Perm1>,<Perm2>', comma separated list of all permissions on Database level. Example: 'Select,Execute,Alter,View definition'
@SubRoles	String	No	'<SubRole1>,<SubRole2>', comma separated list of all subroles. Example: 'db_datareader,db_datawriter,db_ddladmin'

Creates the database role <Role> if it does not already exist. Existing DBPermissions and Subroles are removed and replaced with @DBPermissions and @SubRoles.

QGrip-Remote-Tmp-Usp-Role-Drop

Parameter	Type	Mandatory	Description
@Role	String	Yes	'<Role>', the name of the Role

Drops the database role <Role> if it exists and is a user defined role.

QGrip-Remote-Tmp-Usp-Role-DropAll

Parameter	Type	Mandatory	Description
none			

Drops all existing user defined roles in the database after removing their members.

QGrip-Remote-Tmp-Usp-Role-AddMember

Parameter	Type	Mandatory	Description
@Role	String	Yes	'<Role>', the name of the Role
@Member	String	Yes	'<Member>', the name of the Member. Can be a Database user or Role

Adds Member to Database Role

QGrip-Remote-Tmp-Usp-Role-DropMember

Parameter	Type	Mandatory	Description
@Role	String	Yes	'<Role>', the name of the Role

@Member	String	Yes	'<Member>', the name of the Member. Can be a Database user or Role

Drops Member from Database Role.

6.1.2 QGrip-Remote-Tmp-Usp-DBUser-Create

When you use the QGrip-Remote-Tmp-Usp-DBUser-Create procedure and the accompanying Login does not yet exist, the Login will be created on the Instance. If it is a SQL-Login, the Password Safe will be updated.

If not specified, the following default settings will apply.

Parameter	Default Setting
DBUser	Login name will be used as Database User
DefaultSchema	Dbo
Default Database	Tempdb

QGrip-Remote-Tmp-Usp-DBUser-Create (SQL-Login)

Parameter	Type	Mandatory	Description
@Login	String	Yes	'<SQL-Login>', the name of the SQL-Login
@Integrated	String	Yes	'N', has to be N for a SQL Login.
@DBRoles	String	No	'<Role1>,<Role2>', a comma separated list of all Database Roles the database user should have.

QGrip-Remote-Tmp-Usp-DBUser-Create (Windows-Group)

Parameter	Type	Mandatory	Description
@Login	String	Yes	'<AD-Group>', the name of the AD-Group. The AD-Group must already exist on Domain otherwise creation will fail.
@Integrated	String	Yes	'Y', has to be Y for an AD-Group
@DBRoles	String	No	'<Role1>,<Role2>', a comma separated list of all Database Roles the database user should have.

QGrip-Remote-Tmp-Usp-DBUser-Create (Windows-User)

Parameter	Type	Mandatory	Description
@Login	String	Yes	'<AD-User>', the name of the AD-User. The AD-User must already exist on Domain otherwise creation will fail.
@Integrated	String	Yes	'Y', has to be Y for an AD-User
@DBRoles	String	No	'<Role1>,<Role2>', a comma separated list of all Database Roles the database user should have.

QGrip-Remote-Tmp-Usp-DBUser-Create (Complete)

Parameter	Type	Mandatory	Description
@Login	String	Yes	The name of the Login which can be SQL-Login, AD-Group or AD-User but must be in right combination with the parameter @Integrated. The AD Group/User must already exist on Domain otherwise creation will fail.
@Integrated	String	Yes	'N' for SQL-Login, 'Y' for AD Group/User.

@DBRoles	String	No	'<Role1>,<Role2>', a comma separated list of all Database Roles the database user should have.
@DBUser	String	No	The name of the Database User in case it needs to be different to Login.
@DefaultSchema	String	No	Default Schema
@DefaultDB	String	No	Default database
@ServerRoles	String	No	'<SrvRole1>,<SrvRole2>', a comma separated list of all Server Roles the Login should have.

6.1.3 QGrip-Remote-Tmp-Usp-DBUser-Drop(All)

Note
When you drop a Database User in a PostClone script using this procedure, the accompanying Login will never be dropped.

QGrip-Remote-Tmp-Usp-DBUser-DropAll

Parameter	Type	Mandatory	Description
none			

Drops all (user defined) Database users in the database.

QGrip-Remote-Tmp-Usp-DBUser-Drop

Parameter	Type	Mandatory	Description
@DBUSer	String	Yes	'<DB-User>', the name of the Database User

Drops the Database User <DB-User> if it exists in the current database.

7 Problems dropping Database Users

In a perfectly designed database, objects should only be owned by dbo, a schema or a database role. A Database User should not own any objects. However, in reality this does happen, and therefore consideration needs to be given to stranded objects resulting from a dropped Database User.

When using the QGrip-Remote-Tmp-Usp-DBUser-Drop stored procedure to drop a database user, empty schemas owned by the Database User are automatically dropped.

Message	Possible Solution
ERROR: Drop Schema … Not Possible Message: Schema not Empty	Drop objects in Schema if not used. Transfer owner ship Schema to dbo.
ERROR: Drop Schema … Failed Message: SQL Server Message	Depends on the SQL Server Message
ERROR: Drop Database User Not Possible Message: Used in EXECUTE AS context	Change the procedure/function and choose other construction than EXECUTE AS. (*)
ERROR: Drop Database User Failed Message: SQL Server Message	Depends on the SQL Server Message but most likely owner of objects. Transfer owner ship?

(*) Work around: See below

7.1 Work around: EXECUTE AS

Recreate procedure/function with NewName and without EXECUTE AS clause. Create procedure/function with OldName and with EXECUTE AS clause that only calls NewName.

Old situation

```
CREATE PROCEDURE [dbo].[OldName]
WITH EXECUTE AS 'SQLLogin_3002_P_Pushy_Application'
AS
BEGIN
   /*** Do complicated stuff ***/
END
GO
```

Create new situation

```
DROP PROCEDURE [dbo].[OldName]
GO
CREATE PROCEDURE [dbo].[NewName]
AS
BEGIN
   /*** Do complicated stuff ***/
END
GO
CREATE PROCEDURE [dbo].[OldName]
WITH EXECUTE AS 'SQLLogin_PH_P_Pushy_Application'
AS
BEGIN
   EXEC [dbo].[NewName]
END
GO
```

In the PostClone Script

```
/***********************************************
* Drop Stored Procedure with EXECUTE AS Context
***********************************************/
DROP PROCEDURE [dbo].[OldName]
GO
/***********************************************
* Drop all Database Users
***********************************************/
EXEC [dbo].[Remote-Tmp-Usp-DBUser-DropAll]
GO
/***********************************************
* Create Application User
***********************************************/
   EXEC [dbo].[Remote-Tmp-Usp-DBUser-Create]
     @Login         = 'SQLLogin_PH_{E}_Pushy_Application'
   , @Integrated    = 'N'
   , @DefaultSchema = 'dbo'
   , @DefaultDB     = 'tempdb'
   , @DBRoles       = 'RoleApplication'
GO
/***********************************************
* Create Procedure with EXECUTE AS Context
***********************************************/
CREATE PROCEDURE [dbo].[OldName]
WITH EXECUTE AS 'SQLLogin_PH_{E}_Pushy_Application'
AS
BEGIN
   EXEC [dbo].[NewName]
END
GO
```

8 Password Safe update

If a SQL login is created in a PostClone script, it is created with a generated password of length 24 known to nobody. QGrip will scan the logfiles and search for the certain string

- SYSTEM-INFO1: {XXXX}

If found, the Password Safe will be checked to see if the Password is available using the following criteria

1. Same Instance, Same Application, Same <LoginName>
2. Other Instance, Same Application, Same (DTAP) Environment, Same <LoginName>.

If a match is found in 1 or 2, and the Password Status = OK, the Password will be used to change the Password on the Instance. Otherwise, a Password will be generated and used to change the Password on the Instance.

2 implies that if you move a database from one instance to another instance within the same environment (Develop->Develop, Test->Test, Etc.) the password of logins will be copied as well.

The System info string (SYSTEM-INFO1: {XXXX}) will be removed from the logfile when processed and will not be visible in the Clone logfile.

AlwaysOn
If a database is cloned to an AlwaysOn database, the post clone script will only run on the Primary replica. If a login is created on in the Post Clone script, QGrip will automatically add the login on all replicas with the same SID and password if applicable.