FAQs

What are the server requirements?

Please see the Requirements and Limitations Section.

What is the release time frame?

Please see the Roadmap.

How will SQL Clone be sold?

Initially SQL Clone will be a standalone offering.

Do clones have the same data as the source?

All clones created will have the same data as when the image was taken from the source - it's a 2-stage process. A data image is taken from the live source or a backup (this takes about the same time as a backup). Multiple clones can then be created, each using the image (which can be on a file share) and a local differencing disk.

Can a clone be used like a real database?

Yes, the clone is a real database in every sense - it just has most of the mdf and ldf files stored on a network share (i.e where the snapshot is). Obviously this means the network link's stability and latency will affect the database's performance.

How do I check no one in the team is using a clone before I delete it?

This isn't currently supported within the system, the Activity Monitor in SSMS (or running `sp_who`) can tell you about current usage of a database.

How do I check no one in the team is using an image before I delete it?

Currently, data images can only be deleted when it has no clones created from it. In fact, you will only be able to see the delete icon for a data image on Web client if this is the case.

Can the application clean up failed images?

The application does actually attempt to do this but it currently fails. We are looking to fix this.

Can I identify which databases are clones (so I don't back them up for example)?

When SQL Clone creates a clone database, it adds an Extended Property to the database object named as "IsSQLCloneDatabase" with the value "1". You can take advantage of this mark to find all clone databases under a SQL Server instance, for example with the below query:

Selecting clone databases

```
CREATE TABLE #TempCloneDatabases (DatabaseName VARCHAR(MAX));

INSERT INTO #TempCloneDatabases EXEC sp_MSForEachDB 'Use [?];
SELECT db.name AS DatabaseName
FROM sys.extended_properties AS prop
JOIN sys.databases AS db ON db.name = DB_NAME()
WHERE prop.[class_desc] = "DATABASE" AND prop.name = "IsSQLCloneDatabase" AND prop.value = 1';

SELECT DISTINCT * FROM #TempCloneDatabases ORDER BY DatabaseName;
DROP TABLE #TempCloneDatabases;
```

Will open transactions that are not committed be part of the image file?

Open transactions will not be applied, the clone will have the uncommitted state.

Can images be made from log shipped databases in standby or read-only mode?

The image creation process involves copying .mdf and .ldf files, so it shouldn't pose any problems. Also the image would be in the same state as the database it was created from.

How can I change the default storage location of the clones?

By default, clones are stored under the local app data folder of the account which the SQL Clone Agent service is running under (precisely "%localappdata%\Red Gate\SQl Clone\clones"). You can override the default clones storage folder during agent installation. However, note that this should only be used at initial installation time since it will break any existing clones.

Does the cloning process pick up replication topology?

We expect this to be problematic, but we haven't tested this yet.

Is it possible to use SQL Clone from the command line?

It's currently not available but PowerShell support is in our Roadmap to currently be released as part of SQL Clone 1.0 (as you can see on SQL Clone Roadmap).

Is there a plan for SQL Clone to support password protection when working with SQL Backups?

It's currently a little way down our backlog - the more it's requested, the sooner it will happen.

Can an image from SQL Server 2016 be cloned onto an older server version?

We don't have any near-term plans to provide database downgrade functionality within this product.

It should be possible to achieve this in a script combining our SQL Compare command line, BCP and the Powershell scripting in SQL Clone.

Is there a version with a 32bit C++ redistributable available?

Not currently. The 'instant' technology SQL Clone uses is 64-bit only.

How can I upgrade from SQL Clone technical preview version to beta?

SQL Clone beta is completely incompatible with technical preview version and there is no migration path available. The suggested way here is to delete all the snapshots and clones, uninstall SQL Clone technical preview and install SQL Clone beta.

Are there any features missing from technical preview version?

Yes, there are two key features missing in SQL Clone beta that were available in technical preview version:

- PowerShell Cmdlets - on our Roadmap for SQL Clone 1.0
- A data image cannot be created from a clone database - will likely follow in 1.x, please contact us if you need this functionality