What is a contained database?

A contained database, as the name implies, is a self-contained database that has its own database setting, configurations, and metadata and does not depend on the SQL Server instance

Advantages of a contained database

Having this feature allows the database to bypass having a SQL login to access the database and only depends upon the creation of a SQL User to access the database

Permits the database to be moved to another server without creating ‘security issues’ and orphan SQL Logins

A fully contained database has no configuration dependencies on the instance of the SQL Server (SQL Server 2016)

Partially contained databases make it easier to separate a database from the instance of SQL Server and other databases (SQL Server 2014)

By reducing the ties to the instance of SQL Server, partially contained databases can be useful during failover when you use AlwaysOn Availability Groups.

Creating contained users enables the user to connect directly to the contained database. This is a very significant feature in high availability and disaster recovery scenarios such as in an AlwaysOn solution. If the users are contained users, in case of failover, people would be able to connect to the secondary without creating logins on the instance hosting the secondary. This provides an immediate benefit.

Partially contained databases cannot use replication, change data capture

Steps for creating a contained database via TSQL and GUI

* Before a contained database can be created, it must be enabled using the sp\_configure sproc
* Create a Contained Database using TSQL
* Create an SQL Server User using TSQL

Terminology:

* Contained
  + An element that exists entirely in the database boundary.
* Uncontained
  + An element that crosses the database boundary.
* Non-contained database
  + A database that has containment set to NONE. All databases in versions earlier than SQL Server 2012 are non-contained. By default, all SQL Server 2012 and later databases have a containment set to NONE.

|  |
| --- |
|  |

|  |
| --- |
| /\*  The following demonstration will show that a regular database  which has SQL users and SQL logins DO NOT migrate to another instance  of a database, but rather become an SQL login orphan. But a contained database  when moved, retains the SQL user and SQL login information so as to prevent orphans  \*/  --create test db  CREATE DATABASE PROD  GO  --create sql login and sql user for prod db  USE [master]  GO  CREATE LOGIN [ANDY] WITH PASSWORD=N'password#123',  DEFAULT\_DATABASE=[PROD],  CHECK\_EXPIRATION=OFF, CHECK\_POLICY=OFF  GO  USE [PROD]  GO  CREATE USER [ANDY] FOR LOGIN [ANDY]  GO  USE [PROD]  GO  ALTER ROLE [db\_owner] ADD MEMBER [ANDY]  GO  --backup prod database with the SQL login Andy. Change connection in this query pane to run the restore  Use Master  go  BACKUP DATABASE [PROD]  TO DISK = N'C:\ProdBackup\prod.bak'  WITH COPY\_ONLY,  NOFORMAT,  NOINIT,  NAME = N'PROD-Full Database Backup',  SKIP,  NOREWIND,  NOUNLOAD,  STATS = 10  GO  --Restore database Prod on different instance. Must use a differnt connection to the Dev instance  --Notice that the SQL databse user ANDY moved, but the SQL Login DID NOT copy over!! This is a SQL orphan  USE [master]  RESTORE DATABASE [PROD]  FROM DISK = N'C:\ProdBackup\prod.bak'  WITH FILE = 1,  MOVE N'PROD'  TO N'C:\Program Files\Microsoft SQL Server\MSSQL12.DEV\MSSQL\DATA\PROD.mdf',  MOVE N'PROD\_log'  TO N'C:\Program Files\Microsoft SQL Server\MSSQL12.DEV\MSSQL\DATA\PROD\_log.ldf',  NOUNLOAD, STATS = 5  GO  --Find SQL login orphans. Run this script against the Dev database to find orphans. Change connection in this query pane  USE PROD  EXEC sp\_change\_users\_login 'Report';  --this will fix by mapping the SQL user to the SQL Login  EXEC sp\_change\_users\_login 'Auto\_Fix', 'ANDY', NULL, 'PASSWORD';  ---extra work needed in a non contained database to resole issues of orphans and security!!  use master  go  drop database prod  --drop sql login in both databases  USE [master]  GO  DROP LOGIN [ANDY]  GO  ------------------------------------------------------------------------  --this issue of orphans does not exist in contained databases as  --each database has it's own meta data about security and configuration  --DROP DATABASE ContainDB  --CHANGE THE AUTHENTICATION MODE TO MIXED  --USE [master]  --GO  --EXEC xp\_instance\_regwrite N'HKEY\_LOCAL\_MACHINE',  --N'Software\Microsoft\MSSQLServer\MSSQLServer', N'LoginMode', REG\_DWORD, 2  --GO  ----RESTART SQL SERVER  ----CHANGE THE AUTHENTICATION MODE TO WINDOWS  --USE [master]  --GO  --EXEC xp\_instance\_regwrite N'HKEY\_LOCAL\_MACHINE',  --N'Software\Microsoft\MSSQLServer\MSSQLServer', N'LoginMode', REG\_DWORD, 1  --GO  --RESTART SQL SERVER  -----------------------------------------------------------------------------------------------  --Configure a contained database via sp\_configure  sp\_configure  --set option on  sp\_configure 'contained database authentication', 1  go  reconfigure  go  sp\_configure  --set option off  sp\_configure 'contained database authentication', 0  go  reconfigure  go  sp\_configure  --CREATE A CONTAINED DATABASE. SAME AS CREATING A REGUALAR DATABASE, ONLY THIS TIME ' CONTAINMENT = PARTIAL' ADDEDD  CREATE DATABASE [ContainDB]  CONTAINMENT = PARTIAL --<< ADDITIONAL COMMAND TO CREATE A CONTAINED DATABASE  ON PRIMARY  ( NAME = N'ContainDB',  FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\ContainDB.mdf' ,  SIZE = 4096KB ,  FILEGROWTH = 1024KB )  LOG ON  ( NAME = N'ContainDB\_log',  FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\ContainDB\_log.ldf' ,  SIZE = 1024KB ,  FILEGROWTH = 10%)  GO  --Create a SQL User to access the contained database. Note NO SQL LOGIN!!!  USE [ContainDB]  GO  CREATE USER [Jack] WITH PASSWORD=N'password'  GO  USE [ContainDB]  GO  ALTER AUTHORIZATION ON SCHEMA::[db\_owner] TO [Jack]  GO  --Information about uncontained objects or features. We see that Jack is a SQL USER and  --has database authentication type rather than NONE - which is a reugular database  use ContainDB  go  Select name, type\_desc,authentication\_type\_desc  from sys.database\_principals  --BACKUP THE DATABASE CONTAINDB  Use Master  go  BACKUP DATABASE [CONTAINDB]  TO DISK = N'C:\ProdBackup\CONTAINDB.bak'  WITH COPY\_ONLY,  NOFORMAT,  NOINIT,  NAME = N'CONTAINDB-Full Database Backup',  SKIP,  NOREWIND,  NOUNLOAD,  STATS = 10  GO  --Restore database Prod on different instance. Must use a differnt connection to the Dev instance  --Notice that the SQL databse user ANDY moved, but the SQL Login DID NOT copy over!! This is a SQL orphan  USE [master]  RESTORE DATABASE [CONTAINDB]  FROM DISK = N'C:\ProdBackup\CONTAINDB.bak'  WITH FILE = 1,  MOVE N'CONTAINDB'  TO N'C:\Program Files\Microsoft SQL Server\MSSQL12.DEV\MSSQL\DATA\CONTAINDB.mdf',  MOVE N'CONTAINDB\_log'  TO N'C:\Program Files\Microsoft SQL Server\MSSQL12.DEV\MSSQL\DATA\CONTAINDB\_log.ldf',  NOUNLOAD, STATS = 5  GO  --Find SQL login orphans. Run this script against the Dev database to find orphans. Change connection in this query pane  USE CONTAINDB  EXEC sp\_change\_users\_login 'Report';  --NO ORPHASN FOUND. JACK MOVED WITH THE RESTORE TO THE NEW SERVER  ---EXTRA WORK PREVENTED |
|  |