ASPNETDB Configuration Guide

Introduction

Use this guide along with the **Data Tab Configuration** guide to configure an ASPNETDB-integrated SecureAuth IdP realm.

Prerequisites

1. Have an on-premises **ASPNETDB** data store (see ASP.NET Configuration Steps below to create an ASP.NET database)
2. Designate a service account with read access (and optional write access) for SecureAuth IdP

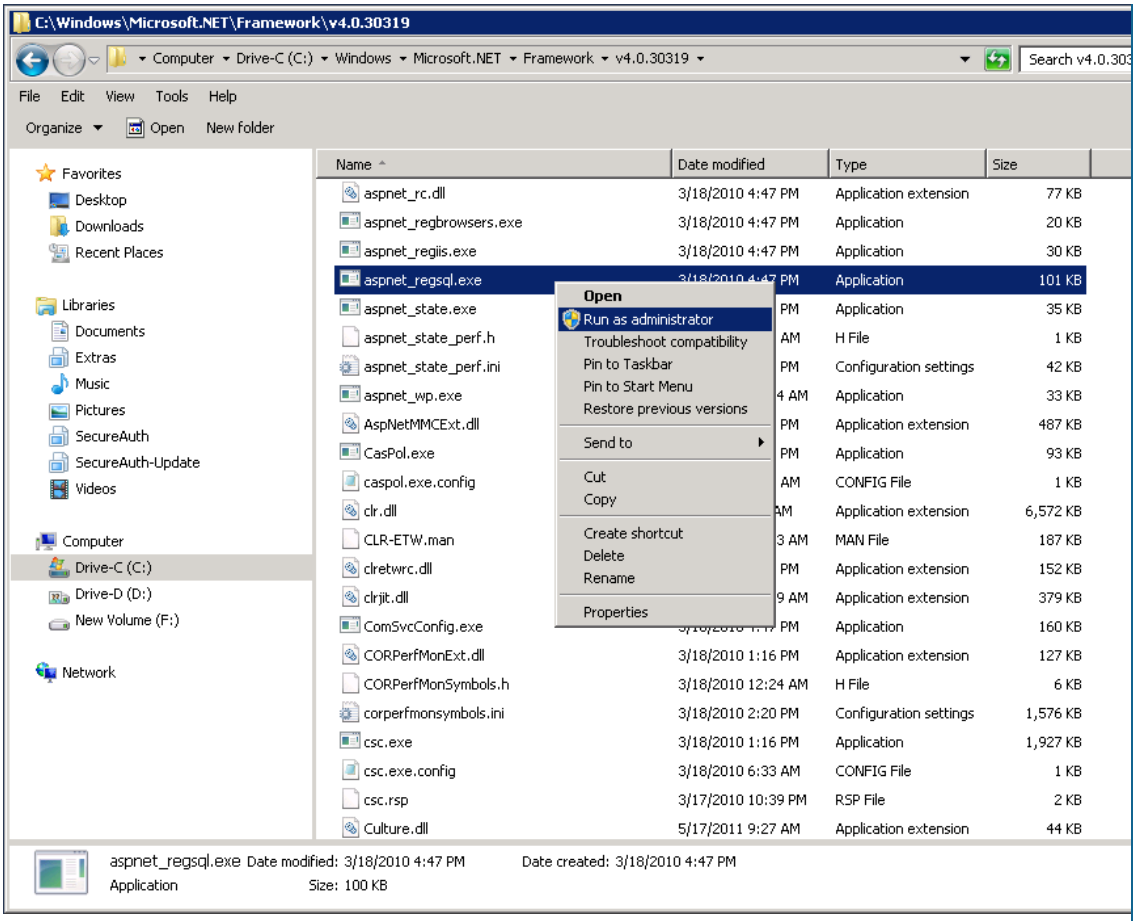
ASP.NET Configuration Steps

Create an ASP.NET Database

Prerequisites

1. Have an asp.net framework
2. Ensure that Microsoft SQL Server is installed on the server
3. Authenticate with a user that has **Create Database** permissions

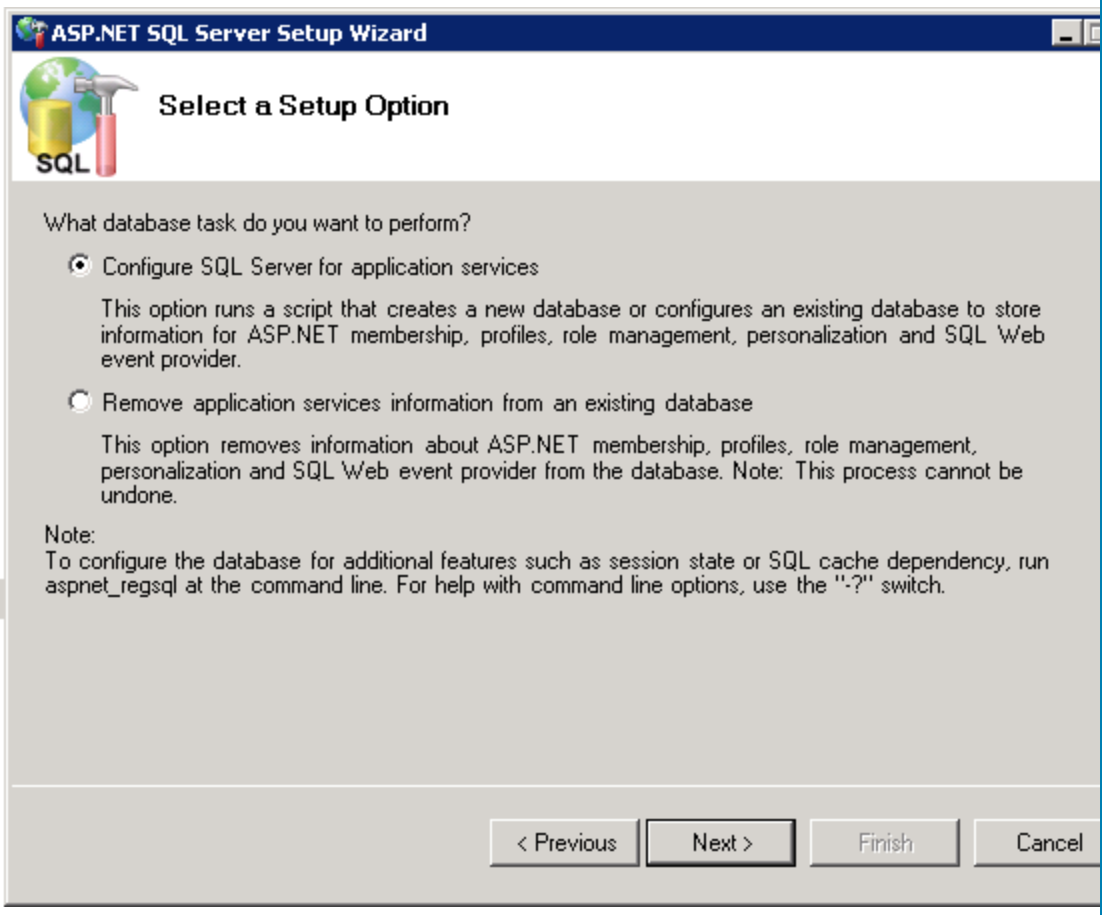
Create the Database



1. Navigate to the following directory on the server: **C:\Windows\Microsoft.NET\Framework\4.0.30319**

2. Locate **aspnet_regsql.exe**, right-click, and select **Run as administrator**

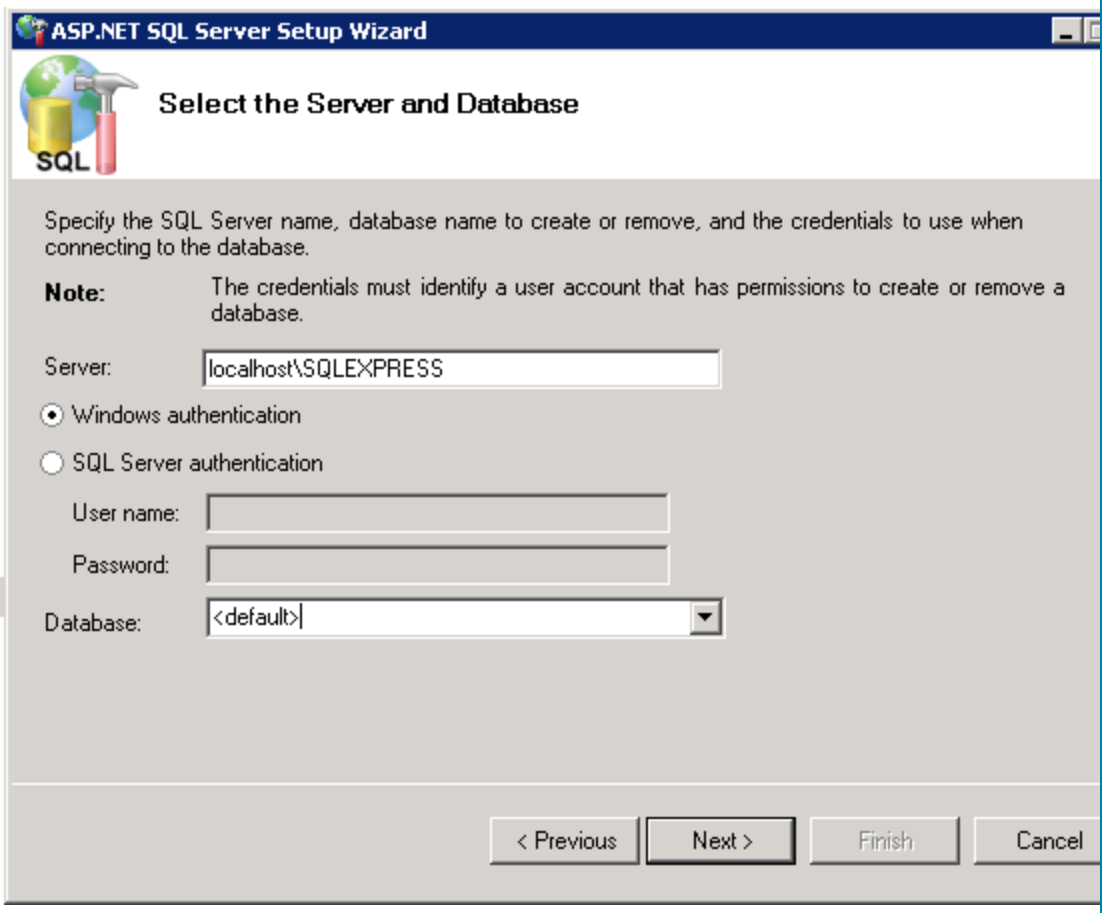
Setup Option



3. On the first prompt (not shown), click **Next**

4. Select **Configure SQL Server for application services** and click **Next**

Server and Database

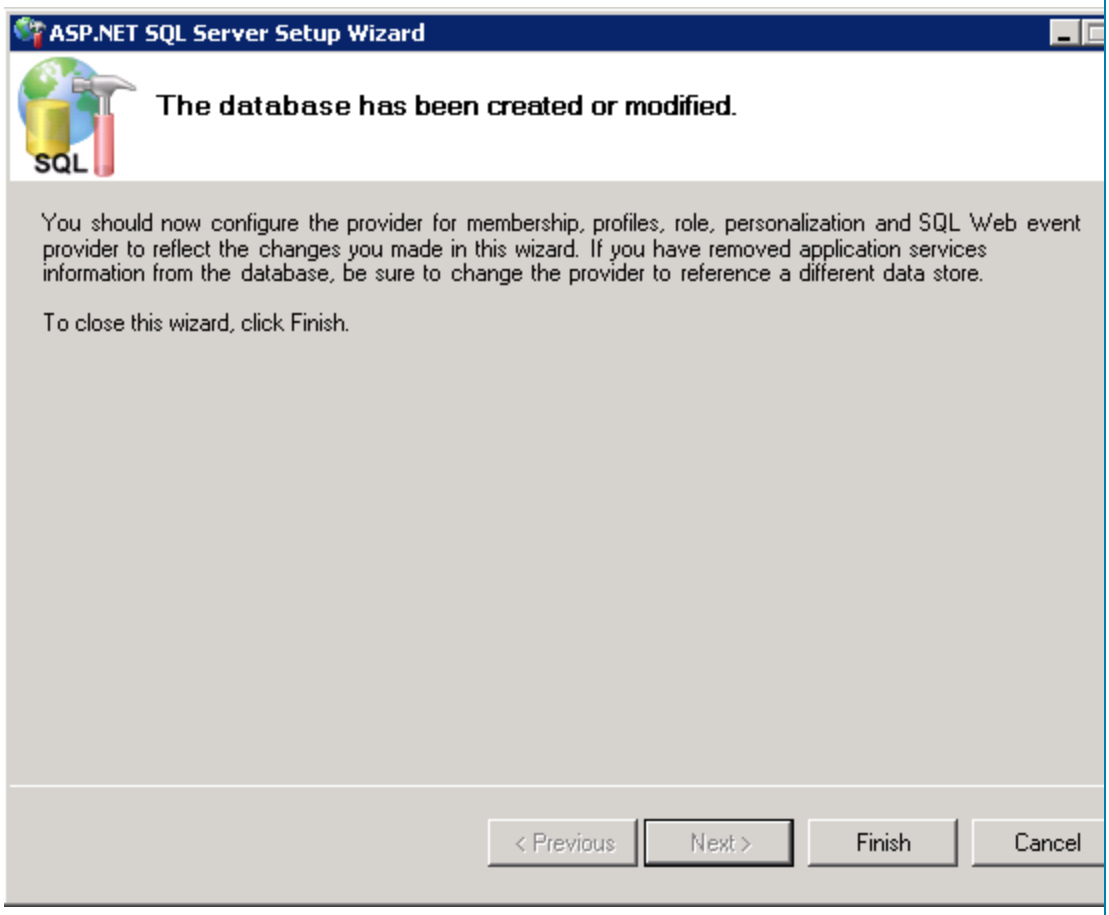


5. Set the **Server** to the server address

6. Select the database name or **<default>** (aspnetdb) from the **Database** dropdown

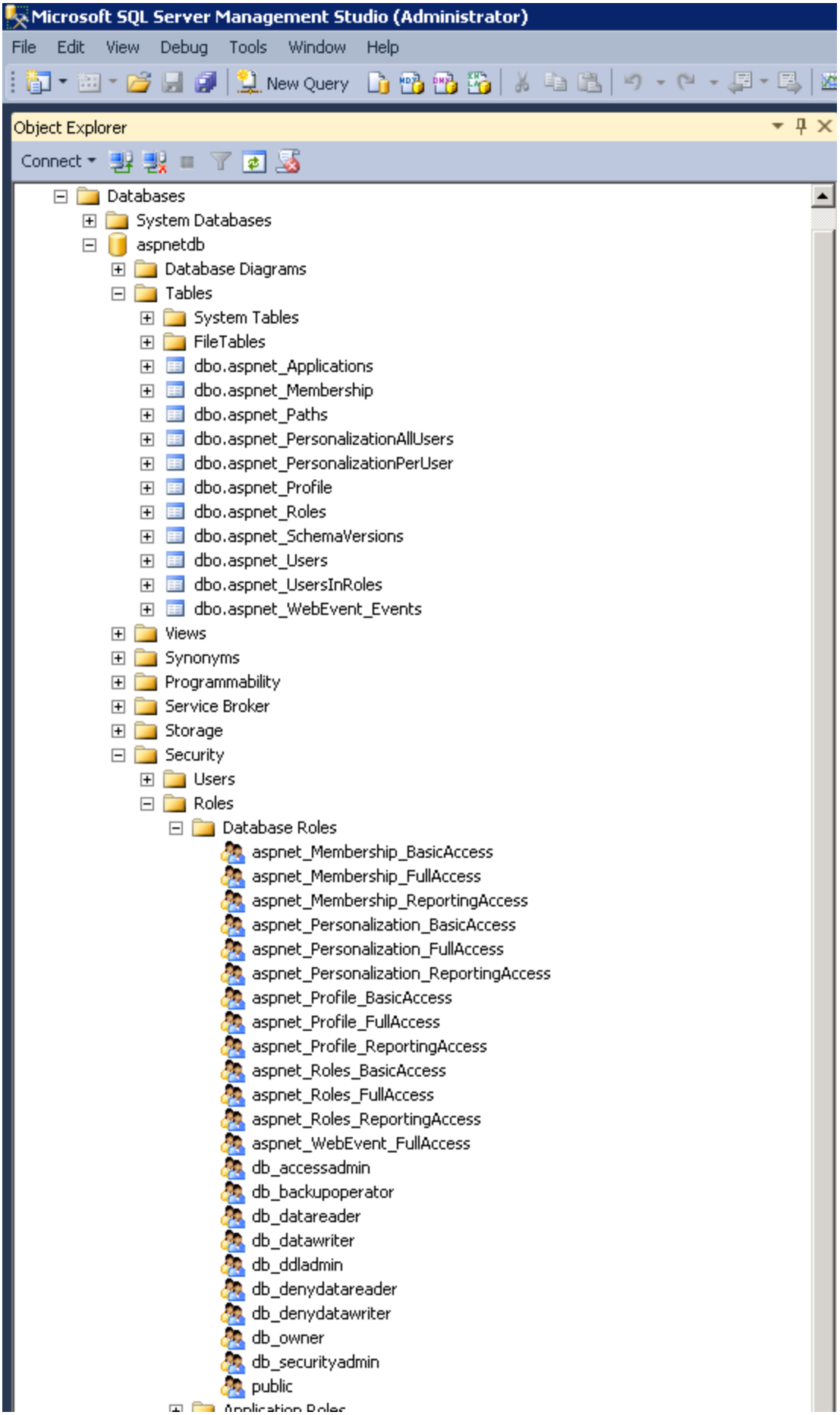
7. Click **Next**

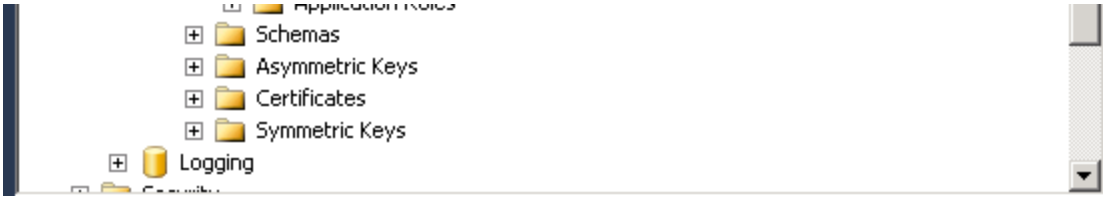
Database Creation



8. Confirm the information (not shown) and click **Next**

9. A success message will appear, and click **Finish** to close the Wizard


Microsoft SQL Server Management Studio (Administrator)
File Edit View Debug Tools Window Help
New Query
Object Explorer
Connect
Databases
System Databases
aspnetdb
Database Diagrams
Tables
System Tables
FileTables
dbo.aspnet_Applications
dbo.aspnet_Membership
dbo.aspnet_Paths
dbo.aspnet_PersonalizationAllUsers
dbo.aspnet_PersonalizationPerUser
dbo.aspnet_Profile
dbo.aspnet_Roles
dbo.aspnet_SchemaVersions
dbo.aspnet_Users
dbo.aspnet_UsersInRoles
dbo.aspnet_WebEvent_Events
Views
Synonyms
Programmability
Service Broker
Storage
Security
Users
Roles
Database Roles
aspnet_Membership_BasicAccess
aspnet_Membership_FullAccess
aspnet_Membership_ReportingAccess
aspnet_Personalization_BasicAccess
aspnet_Personalization_FullAccess
aspnet_Personalization_ReportingAccess
aspnet_Profile_BasicAccess
aspnet_Profile_FullAccess
aspnet_Profile_ReportingAccess
aspnet_Roles_BasicAccess
aspnet_Roles_FullAccess
aspnet_Roles_ReportingAccess
aspnet_WebEvent_FullAccess
db_accessadmin
db_backupoperator
db_datareader
db_datawriter
db_ddladmin
db_denydatareader
db_denydatawriter
db_owner
db_securityadmin
public



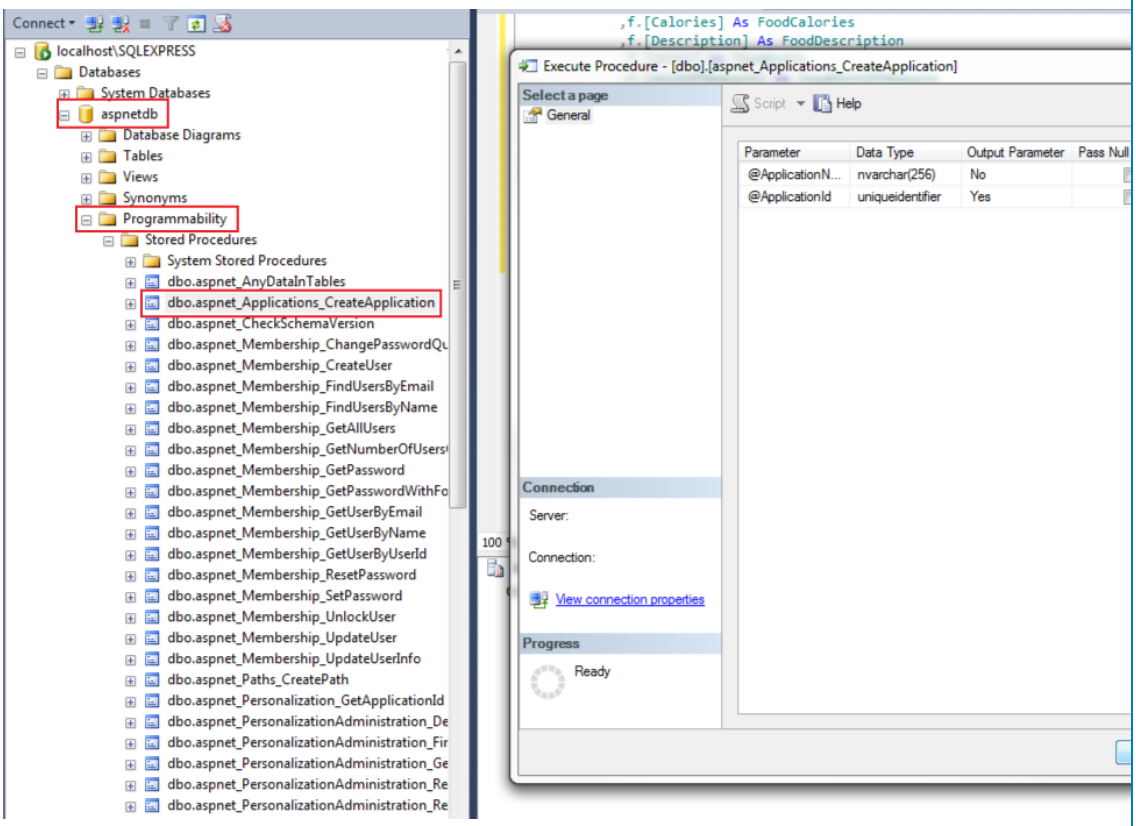
(OPTIONAL) With the new database created, verify that the database, tables, stored procedures, and roles have been created

Update ASPNETDB Schema

10. Download and run the **UpdateSchema.sql** script to update the ASPNETDB schema

    This adds support for Fingerprinting and Push Notifications

Execute Procedure



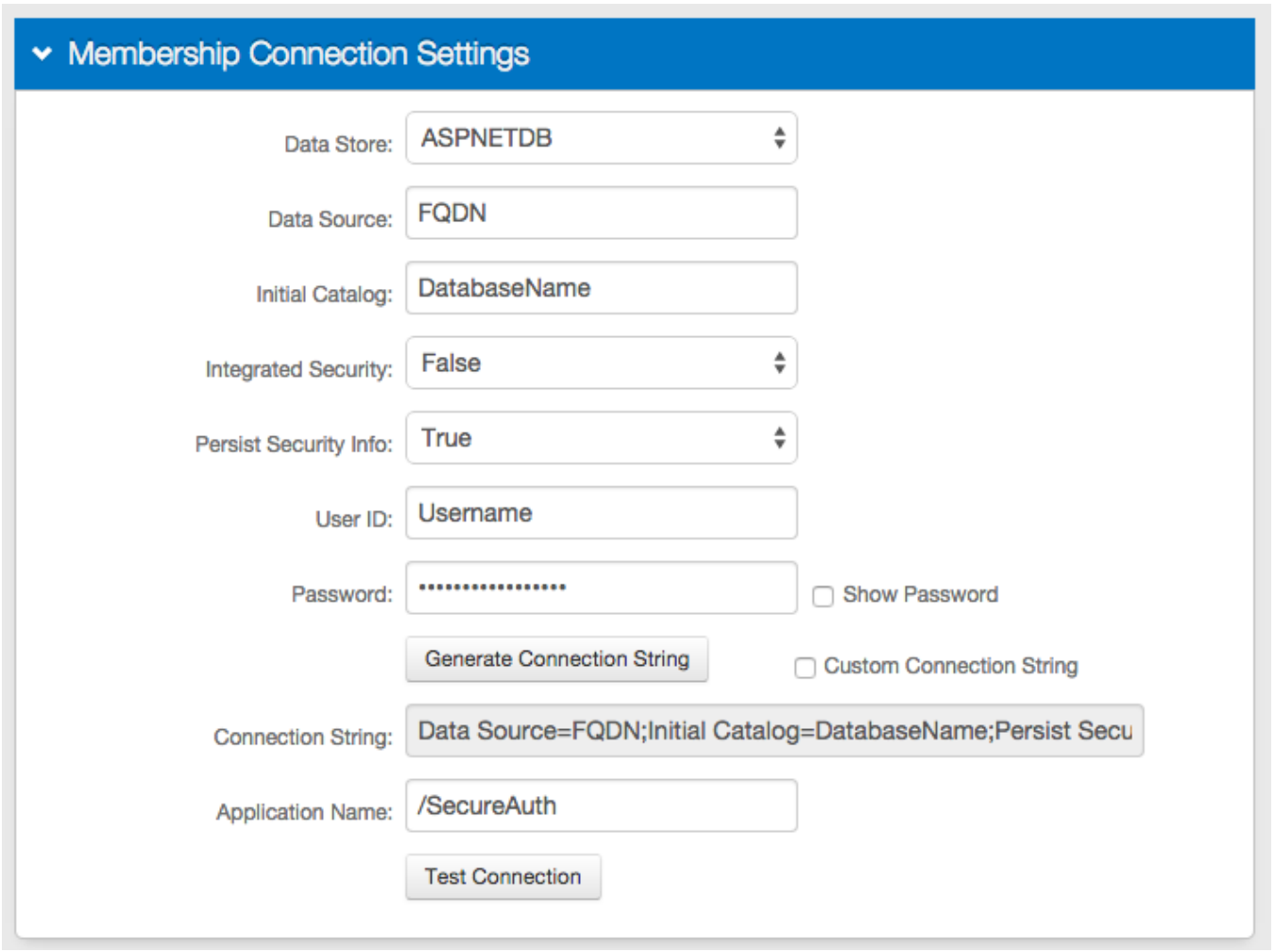
11. In the **aspnetdb**, expand **Stored Procedures** under **Programmability**, and right-click **dbo.aspnet_Applications_CreateApplication**

12. Click **Execute Procedure**, which opens a new window

13. Provide the application name in the **Value** field, e.g. **/SecureAuth**

14. Click **OK**

SecureAuth IdP Configuration Steps


Membership Connection Settings
Data Store:
ASPNETDB
Data Source:
FQDN
Initial Catalog:
DatabaseName
Integrated Security:
False
Persist Security Info:
True
User ID:
Username
Password:
Show Password
Generate Connection String
Custom Connection String
Connection String:
Data Source=FQDN;Initial Catalog=DatabaseName;Persist Secu
Application Name:
/SecureAuth
Test Connection

1. In the **Membership Connection Settings**, select **ASPNETDB** from the **Data Store** dropdown

2. Provide the **Fully Qualified Domain Name (FQDN)** or the **IP Address** in the **Data Source** field

3. Provide the **Database Name** in the **Initial Catalog** field

4. Select **True** from the **Integrated Security** dropdown if the IIS app pool's service account is to be used in the connection (see **Integrated Auth Requirements** below)

    Select **False** to specify an ASPNETDB service account instead

Integrated Auth Requirements

1. Join the server to the domain to utilize a domain service account

2. In IIS, set the application pool **Identity** for both the **.NET v4.5** and **SecureAuth0** app pools to use the preferred service account; and set **Load User Profile** to **True**

3. Make the service account a member of the local administrators group of the SecureAuth IdP server(s)

4. Perform an **IIS reset** after making the changes

5. Select **True** from the **Persist Security Info** dropdown if access to the username and password information is allowed

6. Provide the **User ID** of the SecureAuth IdP Service Account (if **False** is selected in step 4)

7. Provide the **Password** associated to the **User ID** (if **False** is selected in step 4)

8. Click **Generate Connection String**, and the **Connection String** will auto-populate

9. Provide the **Application Name** set in step 13 of the ASP.NET Configuration Steps, e.g. **/SecureAuth**

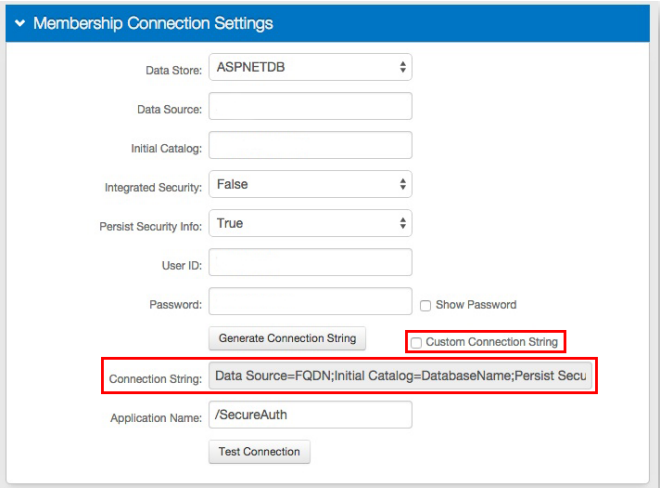
10. Click **Test Connection** to ensure that the connection is successful

If using a **Custom Connection String** and experience an error when testing the connection, then refer to the **Custom Connection String Error** section below for a workaround

Refer to **Data Tab Configuration** to complete the configuration steps in the **Data** tab of the Web Admin

Troubleshooting / Common Issues

Custom Connection String Error

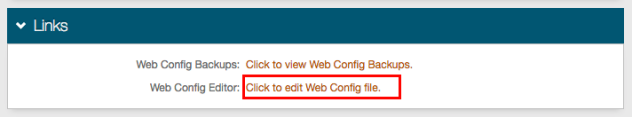


If manually entering a custom connection string, an error may occur when testing the connection, which hinders the ASP.NET Database to successfully integrate with SecureAuth IdP

This error may occur only if **Custom Connection String** is checked, the **Connection String** is manually entered into the field rather than generated by the Web Admin, and the fields that comprise the generated **Connection String** are left empty / default

Workaround

System Info



1. In the **Links** section, select **Click to edit Web Config File**

Web Config Editor

2. Search for **ASPNETDB** and manually enter the connection string into the web.config file

3. Click **Save**

> This enables a successful connection; however, clicking **Test Connection** in the **Data** tab may still yield an error

XML Error

If an XML error occurs while attempting to call **setpropertyvalues**, then the **clientmembership** table and stored procedure may need to be created; and the **aspnet_Profile_FullAccess** role may need to be assigned to the stored procedure

ClientMembership Table and Stored Procedure

ClientMembership

```
CREATE TABLE [dbo].[ClientMembership](
        [UserId] [uniqueidentifier] NULL,
        [ClientGuid] [uniqueidentifier] NULL,
        [CreatedOn] [datetime] NOT NULL,
        [CreatedBy] [varchar](50) NULL,
        [UpdatedOn] [datetime] NULL,
        [UpdatedBy] [varchar](50) NULL
) ON [PRIMARY]

GO

SET ANSI_PADDING OFF
GO

ALTER TABLE [dbo].[ClientMembership] ADD  CONSTRAINT [DF_ClientMembership_CreatedOn]  DEFAULT (getdate())
FOR [CreatedOn]
GO

CREATE PROCEDURE [dbo].[getClientMembership]
    @ApplicationName      nvarchar(256),
    @UserName             nvarchar(256)
AS
BEGIN
        DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@ApplicationName) =
LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN

        DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL

        SELECT @UserId = UserId
    FROM   dbo.aspnet_Users
    WHERE  ApplicationId = @ApplicationId AND LoweredUserName = LOWER(@UserName)

        IF (@UserId IS NULL)
        RETURN

        SELECT Top 1 ClientGuid FROM ClientMembership WHERE @UserID = UserId
END
GO
```

After adding the table and stored procedure (above), update the profile section in the SecureAuth IdP **web.config** file to include a **ClientGUID** property value

web.config profile section

```
<properties>
  <add name="FirstName" />
  <add name="LastName" />
  <add name="AuxID1" />
  <add name="AuxID2" />
  <add name="AuxID3" />
  <add name="AuxID4" />
  <add name="AuxID5" />
  <add name="AuxID6" />
  <add name="AuxID7" />
  <add name="AuxID8" />
  <add name="AuxID9" />
  <add name="AuxID10" />
  <add name="Email1" />
  <add name="Email2" />
  <add name="Phone1" />
  <add name="Phone2" />
  <add name="Phone3" />
  <add name="Phone4" />
  <add name="kbq1" />
  <add name="kbq2" />
  <add name="kbq3" />
  <add name="kba1" />
  <add name="kba2" />
  <add name="kba3" />
  <add name="CertCount" />
  <add name="CertResetDate" />
  <add name="GroupList" />
  <add name="pinHash" />
  <add name="MobileResetDate" />
  <add name="MobileCount" />
  <add name="CertSerialNumber" />
  <add name="ExtSyncPwdDate" />
  <add name="HardwareToken" />
  <add name="iOSDevices" />
  <add name="Email3" />
  <add name="Email4" />
  <add name="OATHSeed" />
  <add name="DigitalFP" type="object" />
          <add name="ClientGUID" />
</properties>
```