Użytkownicy, uprawnienia, role w SQL Server
(W oparciu o SQL Server 2008R2 Books Online)

Tożsamość i kontrola dostępu

- Principals (byty żądające zasobów np. użytkownicy baz danych, konta logowania)
- Securables (obiekty, które mogą być chronione, np. tabele, procedury, funkcje)
- Roles (stałe role serwera, role bazy danych, role użytkowników)
- Permissions (przywileje, które prinicpal otrzymuje do danego securable)

Principals

- Domenowy login systemu Windows
- Lokalny login systemu Windows
- Grupa w systemie Windows
- Login SQL Server
- Użytkownik bazy danych
- Rola bazy danych
- Rola aplikacji

Tryby uwierzytelniania w SQL Server (authentication modes)

Uwierzytelnienie to sprawdzanie tożsamości.

- Uwierzytelnianie Windows
- Tryb mieszany (uwierzytelnianie Windows i uwierzytelnianie SQL Server)
- Tryb uwierzytelniania ustawiamy w Server properties-Security

Tworzenie kont logowania SQL Server

- Natywny login SQL Server

```
CREATE LOGIN testLogin
WITH PASSWORD='1qaz@WSX' MUST_CHANGE,
     DEFAULT_DATABASE=demoDB,
     CHECK_EXPIRATION=ON, CHECK_POLICY=ON
```

MUST_CHANGE – wymusza zmianę hasła przy pierwszym logowaniu
DEFAULT_DATABASE – określa domyślną bazę dla loginu (domyślną bazą jest master)
CHECK_EXPIRATION – wymusza politykę związaną z wygasaniem hasła (domyślnie OFF)
CHECK_POLICY – wymusza politykę systemu Windows stosowaną dla haseł (domyślnie ON)
MUST_CHANGE => CHECK_EXPIRATION + CHECK_POLICY

- Login na bazie konta lokalnego w systemie Windows

```
CREATE LOGIN [c109_21\winLogin] FROM WINDOWS
```

- Login na bazie konta domenowego w systemie Windows

```
CREATE LOGIN [nazwa_domeny\nazwa_loginu] FROM WINDOWS
```

- Certyfikaty i klucze asymetryczne

Login SQL z opcją SID

- Natywny login SQL Server z opcją SID

```
CREATE LOGIN testLogin
WITH PASSWORD='1qaz@WSX' MUST_CHANGE,
     DEFAULT_DATABASE=demoDB
```

- SID danego loginu jest przydatny przy przenoszeniu bazy z jednej instancji SQL Server do innej instancji
- Bez opcji SID nazwa_sid SQL Server przypisze sam numer SID

Tworzenie użytkownika w bazie danych dla loginu

- Uwierzytelnienie w instancji serwera nie implikuje zazwyczaj możliwości pracy z bazami danych w ramach tej instancji
- Wymagany jest użytkownik w bazie danych dla konta logowania

```
CREATE USER user_name FOR
   {
        LOGIN nazwa_loginu
      | CERTIFICATE nazwa_certyfikatu
      | ASYMMETRIC KEY nazwa_klucza_asymetrycznego
   }
   [ WITH DEFAULT_SCHEMA = nazwa_schmatu ]
```

```
USE [demoDB]
GO
CREATE USER [testUser] FOR LOGIN [testLogin]
GO
SELECT suser_sname() 'Nazwa loginu',
       user_name()   'Nazwa użytkownika'
```

Predefiniowane konta użytkowników

- *dbo* – istnieje w każdej bazie, członek roli bazy danych *db_owner*. Konta logowania będące członkami stałej roli serwera *sysadmin* są mapowane na użytkownika dbo
- *guest* - użytkownik pozwalający uzyskać kontom logowania dostęp do baz danych, w których nie mają użytkownika
- Aktywacja użytkownika guest w danej bazie

```
USE [nazwa_bazy]
GO
GRANT CONNECT TO guest
GO
```

Stałe role bazy danych

- **db_owner** – członkowie są administratorami baz danych
- **db_securityadmin** – członkowie zarządzają uprawnieniami, członkostwem w rolach
- **db_accessadmin** - członkowie zarządzają dostępem kont logowania do baz danych
- **db_backupoperator** – członkowie wykonują kopie zapasowe
- **db_ddladmin** – członkowie mogą wykonać dowolną instrujcję języka DDL w danej bazie
- **db_datawriter** – członkowie mogą wykonać dowolną instrujcję języka DML w danej bazie
- **db_datareader -** członkowie mogą wykonać dowolną instrujcję SELECT w danej bazie
- **db_denydatawriter** – przeciwieństwo **db_datawriter**
- **db_denydatareader** – przeciwieństwo **db_datareader**

Stałe role bazy danych

- **db_owner** – członkowie są administratorami baz danych
- **db_securityadmin** – członkowie zarządzają uprawnieniami, członkostwem w rolach
- **db_accessadmin** - członkowie zarządzają dostępem kont logowania do baz danych
- **db_backupoperator** – członkowie wykonują kopie zapasowe
- **db_ddladmin** – członkowie mogą wykonać dowolną instrujcję języka DDL w danej bazie
- **db_datawriter** – członkowie mogą wykonać dowolną instrujcję języka DML w danej bazie
- **db_datareader -** członkowie mogą wykonać dowolną instrujcję SELECT w danej bazie
- **db_denydatawriter** – przeciwieństwo **db_datawriter**
- **db_denydatareader** – przeciwieństwo **db_datareader**

Stałe role serwera

- **sysadmin** – członkowie są administratorami instancji
- **serveradmin** – członkowie zarządzają opcjami konfiguracyjnymi serwera, mogą wykonać shutdown
- **securityadmin** – członkowie zarządzają dowolnymi kontami logowania, użytkownikami, uprawnieniami, członkowstwem w rolach
- **setupadmin** - członkowie zarządzają dołączonymi serwerami
- **dbcreator** – członkowie zarządzają dowolną bazą w instancji
- **diskadmin** – członkowie zarządzają plikami baz danych
- **bulkadmin** – członkowie mogą uruchomić instrukcje BULK INSERT
- **processadmin** - członkowie zarządzają procesami instancji
- **public** – członkowie widzą bazy danych w instancji

Dodawanie loginu do stałej roli sewera realizuje procedura:

sp_addsrvrolemember '*login' , 'rola'*

sp_dropsrvrolemember usuwa członka z roli.

Tworzenie roli w bazie danych

```
CREATE ROLE nazwa_roli
      [ AUTHORIZATION nazwa_właściciela ]

USE demoDB;
CREATE ROLE demoDB_role
            AUTHORIZATION testUser;
GO
```

Domyślnie właścicielem roli będzie użytkownik wydający polecenie CREATE ROLE

Właścicielem roli może być również inna rola

Dodawanie użytkownika/roli do roli realizuje procedura :

sp_addrolemember 'nazwa_roli'
'nazwa_użytkownika/nazwa_roli'

sp_droprolemember usuwa członka z roli

Schematy

- Schemat – kontener składujący obiekty (tabele, widoki, funkcje, procedury, etc)
- Podstawowa składania

```
CREATE SCHEMA nazwa_schematu
                AUTHORIZATION nazwa_właściciela
```

- Składania rozszerzona pozwala tworzyć tabele i widoki wewnątrz schematu i nadawać/odbierać uprawnienia do nich

Przyznawanie/odbieranie/odmawianie uprawnień

```
GRANT uprawnienia ON securable TO prinicpal
REVOKE uprawnienia ON securable TO prinicpal
DENY uprawnienia ON securable TO prinicpal

GRANT/REVOKE/DENY uprawnienia ON pewien_obiekt
                              TO user_login_grupa
```

Polecenie GRANT może zawierać klauzulę WITH GRANT OPTION – principal, który otrzymał dane uprawnienie może je przekazać innym principals
Polecenie DENY odmawia uprawnień. Można zastosować aby wykluczyć uprawnienia nabyte w drodze członkowstawa w grupie, roli
Polecenie REVOKE usuwa uprawnienie przyznane za pomocą GRANT lub odmówione za pomocą DENY

Przyznawanie/odbieranie/odmawianie uprawnień

```
USE demoDB;
GRANT SELECT ON OBJECT::Test_schema.Test_tab1
             TO testUser;
GRANT SELECT ON Test_schema.Test_tab2 TO testUser;
GRANT SELECT ON SCHEMA::dbo TO testUser;
GRANT SELECT ON DATABASE::demoDB TO testUser;
--prawo do uruchamiania procedury
GRANT EXECUTE ON OBJECT::Test_schema.Test_proc1
              TO testUser;
GO

USE demoDB;
REVOKE SELECT ON DATABASE::demoDB TO testUser;
DENY SELECT ON SCHEMA::dbo TO testUser;
GO
```

Typowe przywileje poziomu instancji

- CREATE DATABASE
- ALTER ANY DATABASE
- BACKUP DATABASE
- BACKUP LOG
- CONNECT
- VIEW ANY DEFINITION
- Rola default obejmuje przywileje connect oraz view any definition

Użytkownik dbo

- Każda baza posiada użytkownika dbo
- dbo to administrator bazy danych
- Login sa oraz członkowie roli instancji sysadmin, właściciel bazy są mapowani w każdej bazie na dbo

Łańcuchy właścicielstwa (ownership chains)

- Odwołania do obiektów securable mogą odbywać się za pośredncitwem innych obiektów securable. Ciąg odwołań tworzy tak zwany łańcuch
- SQL Server sprawdza uprawnienia w łańcuchu tylko jeśli zmienia się właściciel dange obiektu securable

Łańcuchy właścicielstwa - przykład

Widok A – właściciel User1,

↓ User 3 ma prawo select tylko na tym widoku

Widok B – właściciel User1

↓ do tego momentu łańcuch właścicielstwa jest nieprzerwany

Tabela C – właściciel User2

Załóżmy, że User3 wykonuje select na Widoku A
- SQL Server zweryfikuje czy User3 ma prawo select na Widok A i potwierdzi, że ma
- Widok A korzysta z B, SQL Server sprawdzi czy A i B ma tego samego właściciela, jeśli tak to nie sprawdza czy User3 ma stosowne uprawnienia do B
- Widok B korzysta z tabeli C – B i C mają innego właściciela, nastąpi sprawdzenie uprawnień

Użytkownicy tworzeni w oparciu o certyfikat

- User1 – właściciel schematu s1
- User2 – właściciel schematu s2
- W s2 jest składowana procedura sp2
- W s1 jest składowana procedura sp1, która wywołuje sp1
- User3 ma prawo execute na sp1, lecz nie ma prawa execute na sp2, wywołanie sp1 zakończy się porażką

Sposób pozwalający wywołać user3 sp1 bez przypisania mu prawa execute do sp2

- Utworzyć klucz główny bazy danych
- Utworzyć certyfikat
- Podpisać procedurę sp1 certyfikatem
- Stworzyć użytkownika bazy danych w oparciu o certyfikat
- Nadać temu użytkownikowi prawo execute do sp2

Problem niezgodnych SID

- Dla loginu generowany jest przez SQL serwer identyfikator tzw. SID
- Użytkownicy dla loginu są również tworzeni w oparciu o ten SID
- Pojawia się problem po dołączeniu (atach) lub odtworzeniu (restore) bazy z jednego serwera na drugi serwer
- Tworząc loginy dla użutkowników baz danych trzeba znać SID loginu
- Inne wyjście – zmodyfikować użytkownika
    - ALTER USER nazwa_uzytkownika WITH LOGIN nazwa_loginu