Sicherheit auf Zeilenebene

Motivation

- Nicht jeder Benutzer soll alle Zeilen lesen dürfen
  - Vertraulichkeit / Schutz von Daten gewährleisten
- Ein Benutzer soll nur „seine“ Daten lesen dürfen
  - Bsp. Krankenhaus → Abteilung soll nur die Daten ihrer eigenen Patienten lesen dürfen
- Vorgehensweise:
  - Erstellen einer Tabelle mit mindestens einem Merkmal des Benutzers für die Filterung
  - Erstellen einer Inline-Funktion
  - Erstellen und aktivieren einer Richtlinie

Eigenschaften

- Filterprädikate filtern Zeilen, welche für Lesevorgänge zur Verfügung stehen
- Blockprädikate blockieren Schreibvorgänge
- Der Zugriff wird über ein Sicherheitsprädikat beschränkt, welches als Inline-Tabellenwertfunktion definiert ist

- Filestream und Polybase sind nicht kompatibel mit RLS (row level security)
  - Unterliegt weiteren Einschränkungen

    https://msdn.microsoft.com/library/bd102e95-53e2-4da6-9b8b-0e4f02d286d3

Beispiel

- Erstellen einer Tabelle

```
CREATE TABLE Testtabelle
(
    ID INT IDENTITY PRIMARY KEY,
    Benutzer sysname NOT NULL,
    Daten NVARCHAR(max)
);
```

- Erstellen verschiedener Benutzer

```
CREATE USER U1 WITHOUT LOGIN;
CREATE USER U2 WITHOUT LOGIN;
CREATE USER Boss WITHOUT LOGIN;
```

  - Die Benutzer-Spalte dient zum Abspeichern des Benutzernamens (sysname siehe msdn)
  - Auf dieser Spalte soll das Filterprädikat arbeiten
  - Benutzer ohne Login für Testzwecke (→ Rollen, Anmeldungen, Benutzer)

- Datensätze einfügen und Benutzern die Leserechte geben

```
-- Leseberechtigung geben
GRANT SELECT ON Testtabelle TO U1
GRANT SELECT ON Testtabelle TO U2
GRANT SELECT ON Testtabelle TO Boss
```

Funktion

- Inline-Funktion soll die Filter definieren

```
CREATE FUNCTION SicherheitsSchema.fn_zeilenfilter(@Username AS sysname)
    RETURNS TABLE
WITH SCHEMABINDING
AS
    RETURN  SELECT
                1 AS fn_sicherheitsergebnis
            WHERE @Username = USER_NAME()
                        OR USER_NAME() = 'Boss'
                        OR USER_NAME() = 'dbo'
;
```

Best practice: Funktion einem eigenen Schema zuordnen

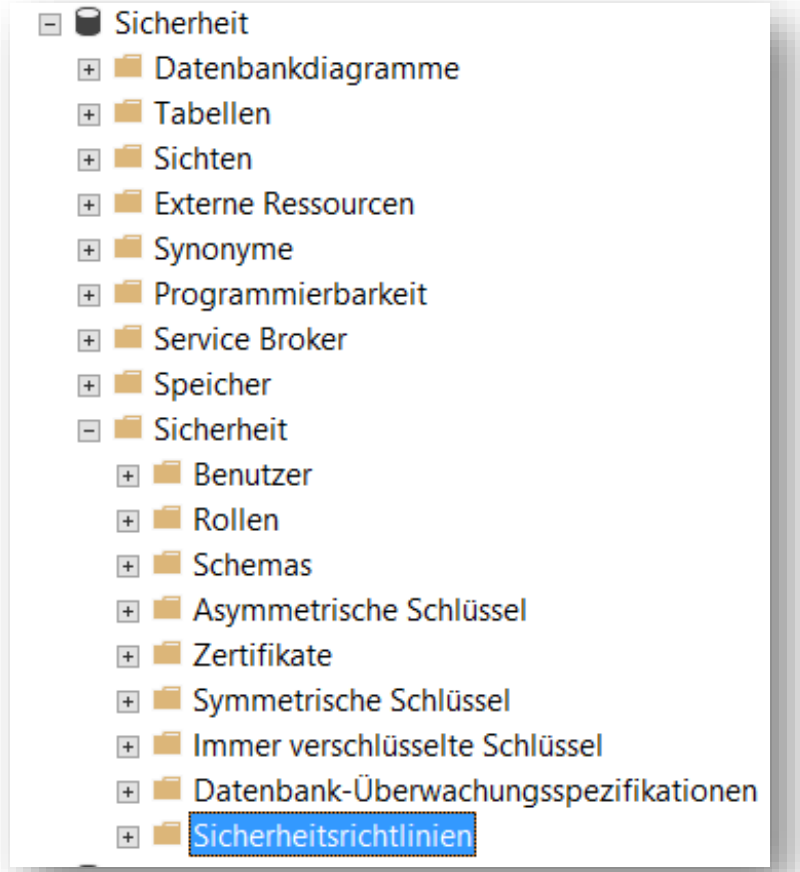
Entscheidend für die Filterung nach Benutzern

  - Die Funktion gibt immer eine 1 zurück, wenn der mitgelieferte @Username mit dem aktuellen Benutzer übereinstimmt, oder der aktuelle Benutzer ***Boss*** oder ***dbo*** ist

- Anschließend Sicherheitsrichtlinie erstellen, welche auf Basis dieser Funktion arbeitet

Sicherheitsrichtlinie

- Wird per T-SQL erstellt
- Nutzt die eben erstellte Funktion als Prädikat
- Im Objekt-Explorer unter Datenbank → Sicherheit




```
CREATE SECURITY POLICY FilterPolicy
    ADD FILTER PREDICATE
        SicherheitsSchema.fn_zeilenfilter(Benutzer)
        ON dbo.Testtabelle
        WITH (STATE = ON);
```

- ➢ Status muss auf ON sein, damit die Richtlinie greift
- ➢ Der Eintrag aus der Benutzerspalte muss dem Prädikat mitgegeben werden