## Модуль 1: Пользователи, роли и ограничение доступа

### Урок 1: Аутентификация подключения к SQL Server

Аутентификация — процесс проверки личности пользователя, который пытается подключиться к SQL Server.

#### Основные элементы безопасности

**Securables (объекты)**:

К которым осуществляется доступ: базы данных, схемы, таблицы.

Иерархия: БД → схема → таблица (как папки внутри папок).

**Principals (субъекты)**:

Кто осуществляет доступ: пользователи, роли (группы пользователей).

Иерархия: роль – роль – пользователь

**Permissions (разрешения)**:

На доступ: чтение (SELECT), запись (INSERT), управление (CONTROL).

Иерархия: CONTROL SERVER – ALTER DATABASE – CREATE DATABASE

#### Виды аутентификации

**Аутентификация Windows**:

Использует учётную запись Windows

**Смешанная аутентификация**:

Поддерживает Windows и логины SQL Server

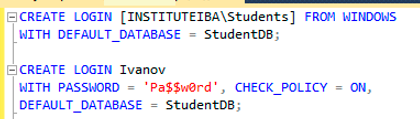
#### Логины и их создание

**Логин** — объект уровня сервера. Он не дает права на доступ к данным бд. Для доступа к данным нужно создать user.

**Создание:**

В **SSMS**: раздел "Logins" → "New Login".

**Через Transact-SQL:**

****

Другие компоненты SQL Server (например, службы) тоже могут создавать логины.

|  |
| --- |
| USE MyDatabase;  CREATE USER Ivan FOR LOGIN Ivan; |

Теперь "Ivan" может работать с "MyDatabase".

#### Специальные пользователи

**dbo (database owner)**:

На пользователя **dbo** отображается логин **sa**, владелец базы данных и все члены встроенной роли sysadmin

**guest**:

На пользователя **guest** отображаются логины без доступа к бд, по умолчанию он отключен.

#### Системные функции

Показывают информацию о текущем пользователе:

SELECT \* FROM sys.user\_token; — токены пользователя в базе.

SELECT \* FROM sys.login\_token; — токены логина на сервере.

#### Автономные базы данных (Partially Contained)

Обычные базы привязаны к серверным логинам, что неудобно при переносе.

Они же не зависят от серверного логина.

Автономные базы хранят пользователей внутри себя:

Поддерживают пользователей Windows или с паролем.

Удобны для миграции.

**Для создания автономной базы данных:**

В SSMS: свойства сервера → "Enable Contained Databases".

|  |
| --- |
| CREATE DATABASE MyContainedDB WITH CONTAINMENT = PARTIAL;  USE MyContainedDB;  CREATE USER Ivan WITH PASSWORD = '12345'; |

### Урок 2: Роли

Роли — группы пользователей, которым можно дать общие права. Это упрощает управление доступом.

Иерархия серверных разрешений: CONTROL – ALTER - CREATE

#### Разрешения уровня сервера

Управляют ресурсами сервера (создание баз, управление настройками).

Разрешение CONTROL SERVER включает в себя все другие разрешения

Это разрешение предоставлено роли **sysadmin**

Иметь разрешение CONTROL SERVER – не то же самое, что быть членом роли **sysadmin**

#### Фиксированные серверные роли

sysadmin — полный контроль над сервером.

serveradmin — управление настройками.

securityadmin — управление логинами.

processadmin — управление процессами.

setupadmin — настройка связанных серверов.

bulkadmin — массовая загрузка данных.

diskadmin — управление дисками.

dbcreator — создание и изменение баз.

public — базовая роль для всех логинов.

#### Пользовательские серверные роли

Создание и удаление

|  |
| --- |
| CREATE SERVER ROLE MyRole;  DROP |

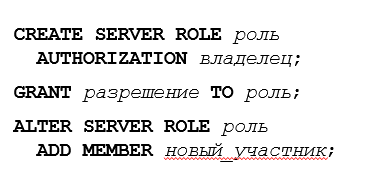
Управление разрешениями (GRANT, DENY, REVOKE)

|  |
| --- |
| GRANT SELECT TO MyRole; |

Управление челенством

|  |
| --- |
| ALTER SERVER ROLE MyRole ADD MEMBER Ivan; |

**Назначение пользовательских серверных ролей:**



#### Разрешения уровня бд

Управляют доступом к ресурсам сервера (чтение таблиц).

CONTROL: Даёт все права в базе (кроме CREATE DATABASE).

Присвоено роли db\_owner.

Некоторые разрешения уровня бд предоставляются автоматически из серверных разрешений

#### Фиксированные роли бд

db\_owner — полный контроль над базой.

db\_datareader — чтение всех данных.

db\_datawriter — запись данных.

db\_securityadmin — управление безопасностью.

db\_accessadmin — управление доступом.

db\_denydatawriter — запрет записи.

db\_denydatareader — запрет чтения.

db\_backupoperator — создание резервных копий.

db\_ddladmin — управление структурой (DDL).

public — базовая роль.

#### Владелец бд

Пользователь dbo — владелец базы: член роли db\_owner.

Назначение владельца:

|  |
| --- |
| ALTER AUTHORIZATION ON DATABASE::MyDatabase TO Ivan; |

#### Пользовательские роли бд

|  |
| --- |
| USE MyDatabase;  CREATE ROLE MyDBRole;  GRANT SELECT ON Table1 TO MyDBRole; (DENY, REVOKE)  ALTER ROLE MyDBRole ADD MEMBER Ivan;  DROP ROLE MyDBRole; |

#### Роли приложения

Заменяют контекст пользователя

|  |
| --- |
| CREATE APPLICATION ROLE AppRole WITH PASSWORD = '123';  GRANT SELECT ON Table1 TO AppRole; |

Активация:

|  |
| --- |
| EXEC sp\_setapprole 'AppRole', '123'; |

Деактивация (с сохранением контекста):

|  |
| --- |
| DECLARE @cookie VARBINARY(8000);  EXEC sp\_setapprole 'AppRole', '123', @cookie OUTPUT;  EXEC sp\_unsetapprole @cookie; |

### Урок 3: Предоставление доступа к объектам

#### Участники безопасности (Principals)

Уровень сервера: логины.

Уровень базы: пользователи, роли.

#### Иерархия защищаемых объектов

Структура: Сервер → База данных → Схема → Объект.

Полное имя: Server.Database.dbo.Object.

#### Команды разрешений на T-SQL

GRANT — разрешить:

|  |
| --- |
| GRANT SELECT ON MyTable TO Ivan; |

DENY — запретить (имеет преимущество):

|  |
| --- |
| DENY SELECT ON MyTable TO Ivan; |

REVOKE — отозвать:

|  |
| --- |
| REVOKE SELECT ON MyTable FROM Ivan; |

#### Примеры разрешений

**Таблицы**: SELECT, INSERT, UPDATE, DELETE, REFERENCES.

**Хранимые процедуры**: EXECUTE, ALTER, VIEW DEFINITION.

**Функции**: EXECUTE (скалярные), SELECT (табличные).

#### Цепочка владельцев

Если объект B (представление, процедура или функция) ссылается на объект A (например, таблица), то:

Если владельцы A и B совпадают, то пользователю достаточно иметь разрешение на объект B

Если владельцы A и B не совпадают, то пользователю нужно иметь разрешение и на объект A, и на объект B

Пример 1:

|  |
| --- |
| *-- Ivanov*  CREATE TABLE T1 (Id INT);  GRANT SELECT ON T1 TO Petrov;  CREATE VIEW V1 AS SELECT Id FROM T1;  GRANT SELECT ON V1 TO Sidorov;  *-- Sidorov видит V1, если T1 и V1 принадлежат Ivanov* |

Пример 2:

|  |
| --- |
| *-- Ivanov*  CREATE TABLE T1 (Id INT);  GRANT SELECT ON T1 TO Petrov;  *-- Petrov*  CREATE VIEW V1 AS SELECT Id FROM T1;  GRANT SELECT ON V1 TO Sidorov;  *-- Sidorov нужен доступ к T1, так как владельцы разные* |

#### Порядок разрешения имён объектов

Если имя объекта не полное:

Проверяет схему sys (для системных процедур).

Схему пользователя по умолчанию.

Схему dbo.

#### Использование схем

Схема — "папка" для объектов:

|  |
| --- |
| CREATE SCHEMA game AUTHORIZATION mmtestapp;  CREATE TABLE game.T1 (Id INT);  GRANT SELECT, INSERT ON SCHEMA::game TO Ivan; *-- Доступ ко всем объектам в схеме* |