جامعة طرابلس
كلية تقنية المعلومات

Advanced Databases قواعد البيانات المتقدمة

IT IS-325

د. عبدالسلام منصور الشريف

a.abdoessalam@uot.edu.ly

المحاضرة الثالثة عشر – لغة التحكم في البيانات II

Data Control Language II

Contents

- Fixed Server and Database Roles
- REVOKE
- DENY

Fixed Server-Level Roles

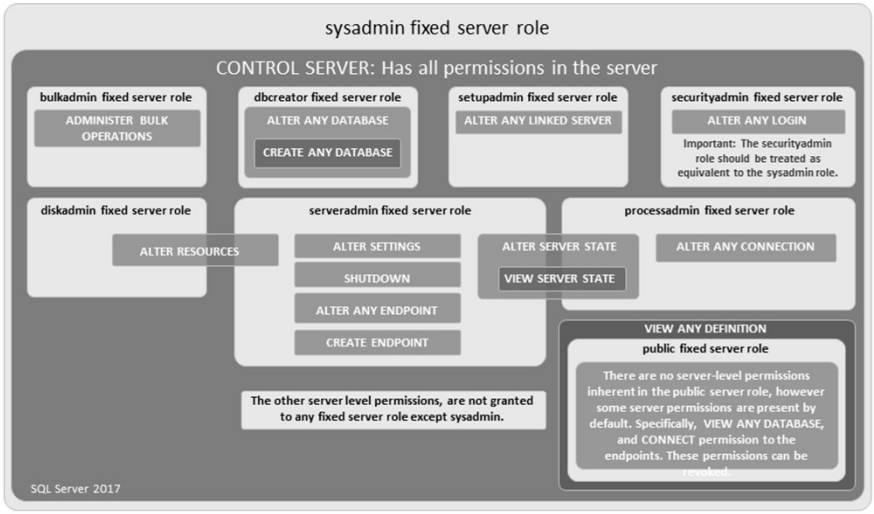
- **The fixed server–level roles and their capabilities**.
- **sysadmin**
  - Members of the sysadmin fixed server role can perform any activity in the server.
- **serveradmin**
  - Members of the serveradmin fixed server role can change server-wide configuration options and shut down the server.
- **securityadmin**
  - Members of the securityadmin fixed server role manage logins and their properties. They can GRANT, DENY, and REVOKE server-level permissions. They can also GRANT, DENY, and REVOKE database-level permissions if they have access to a database. Additionally, they can reset passwords for SQL Server logins.
- **processadmin**
  - Members of the processadmin fixed server role can end processes that are running in an instance of SQL Server.

Fixed Server-Level Roles

- **setupadmin**
  - **Members of the setupadmin fixed server role can add and remove linked servers by using Transact-SQL statements.**
  -
- **bulkadmin**
  - **Members of the bulkadmin fixed server role can run the BULK INSERT statement.**
- **diskadmin**
  - **The diskadmin fixed server role is used for managing disk files.**
- **dbcreator**
  - **Members of the dbcreator fixed server role can create, alter, drop, and restore any database.**
- **public**
  - **Every SQL Server login belongs to the public server role.**

Fixed Server-Level Roles

SERVER LEVEL ROLES AND PERMISSIONS: 9 fixed server roles, 34 server permissions



Manipulating Server Roles

- Add and Remove users from and to Roles.

```
ALTER SERVER ROLE server_role_name
{
    [ ADD MEMBER server_principal ]
  | [ DROP MEMBER server_principal ]
  | [ WITH NAME = new_server_role_name ]
} [ ; ]
```

- **ADD MEMBER *server_principal***
  Adds the specified server principal to the server role. *server_principal* can be a login or a user-defined server role.

- **DROP MEMBER *server_principal***
  Removes the specified server principal from the server role. *server_principal* can be a login or a user-defined server role.

- **WITH NAME =*new_server_role_name***
  Specifies the new name of the user-defined server role. This name cannot already exist in the server.

Manipulating Server Roles

- The following example adds a SQL Server login named Ahmed to the **diskadmin** fixed server role.

```
ALTER SERVER ROLE diskadmin ADD
MEMBER Ahmed ;
```

- The following example removes a SQL Server login named Ahmed from the **diskadmin** fixed server role.

```
ALTER SERVER ROLE diskadmin DROP
MEMBER Ahmed ;
```

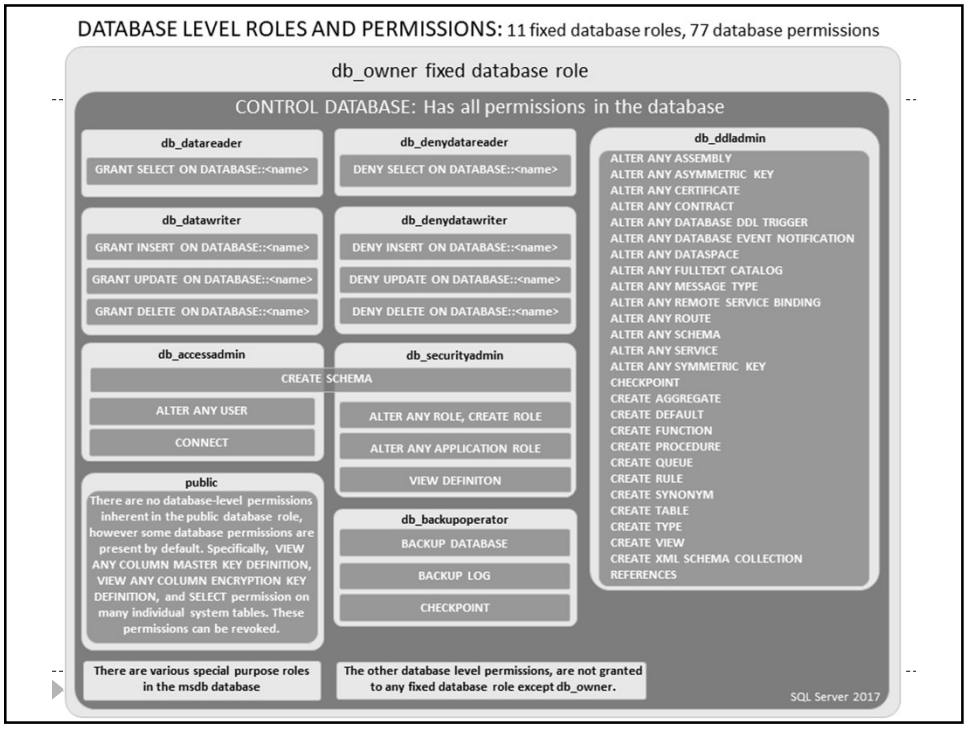
Fixed-database roles

Fixed-database roles and their capabilities. These roles exist in all databases.

- **db_owner**
  Can perform all configuration and maintenance activities on the database, and can also drop the database in SQL Server.
- **db_securityadmin**
  Can modify role membership for custom roles only and manage permissions.
- **db_accessadmin**
  Can add or remove access to the database for Windows logins, Windows groups, and SQL Server logins.

Fixed-database roles

- **db_backupoperator**
  Can back up the database.
- **db_ddladmin**
  Can run any Data Definition Language (DDL) command in a database.
- **db_datawriter**
  Can add, delete, or change data in all user tables.
- **db_datareader**
  Can read all data from all user tables and views.
- **db_denydatawriter**
  Cannot add, modify, or delete any data in the user tables within a database.
- **db_denydatareader**
  Cannot read any data from the user tables and views within a database.



Manipulating Database Roles

- Adds or removes members to or from a database role, or changes the name of a user-defined database role.

```
ALTER ROLE role_name
{
    ADD MEMBER database_principal
  | DROP MEMBER database_principal
  | WITH NAME = new_name
} [ ; ]
```

- **Role_name**
  Specifies the database role to change.
- **ADD MEMBER *database_principal***
  Adds the specified database principal to the database role.
- **DROP MEMBER *database_principal***
  Removes the specified database principal from the database role.
- **WITH NAME =*new_name***
  Specifies the new name of the user-defined database role. This name cannot already exist in the database.

Manipulating Database Roles

- The following example adds the user 'Ahmed' to the fixed database-level role **db_datareader**.

```
ALTER ROLE db_datareader ADD MEMBER Ahmed;
GO
```

- The following example removes a database user 'Ahmed' from the **db_datareader** fixed database role.

```
ALTER ROLE db_datareader DROP MEMBER Ahmed ;
GO
```

How do Permissions work?

- Removes a previously granted or denied permission.

```
REVOKE [ GRANT OPTION FOR ]
       permission [ ,...n ]
       {TO | FROM} database_principal[ ,...n ]
       [ CASCADE]
       [ AS database_principal ]
```

- CASCADE

Indicates that the permission being revoked is also revoked from other principals to which it has been granted.

- AS principal

Use the AS principal clause to indicate that you are revoking a permission that was granted by a principal other than you.

How do Permissions work?

- Revokes EXECUTE permission on the stored procedure `[dbo].[getStudentSemesters]` from user Ahmed.

```
REVOKE EXECUTE ON
OBJECT::[dbo].[getStudentSemesters]
    FROM Ahmed;
GO
```

How do Permissions work?

- Revokes VIEW DEFINITION permission on the `[ITDatabase]` database from user Ahmed and from all principals to which Ahmed has granted VIEW DEFINITION permission.

```
REVOKE VIEW DEFINITION FROM Ahmed CASCADE;
GO
```

How do Permissions work?

- Grant and Revoke Select Permission on pertest

```
CREATE SCHEMA pertest;
GO
CREATE USER Ahmed without login;
GO
CREATE ROLE Managers;
GO
ALTER ROLE Managers ADD MEMBER Ahmed;
GO
GRANT SELECT ON SCHEMA :: pertest TO Managers;
GO
REVOKE SELECT ON SCHEMA :: pertest TO Managers;
GO
```

How do Permissions work?

- Denies a permission to a principal. Prevents that principal from inheriting the permission through its group or role memberships.

```
DENY <permission> [ ,...n ]  }
     TO principal [ ,...n ]
     [ AS principal ]  [;]
```

How do Permissions work?

- Denies VIEW DEFINITION permission on the `[ITDatabase]` database to user Ahmed and to all principals to which Ahmed has granted VIEW DEFINITION permission.

```
DENY VIEW DEFINITION TO Ahmed CASCADE;
GO
```