

# SQL Server 安全主体和安全对象

一般来说,通过给主体分配对象的权限来实现 SQL Server 上的用户与对象的安全。在这一系列,你会学习在 SQL Server 实例中通过权限授权来执行操作及访问安全对象。在 SQL Server 中重要的主体是角色,你会学习角色可以让安全管理比使用单独用户更容易。你也会学习 SQL Server 的安全对象。

## 授权(Authorization)

Authentication is only part of accessing all of the goodies in a database server. 认证(Authentication)有点像是有一个护照证明你是谁,但没有签证。你需要一个签证才能访问其他国家。在这一篇你会学习授权以及它是如何作为签证访问数据库对象。

主体是一个用户或进程可以访问一个或多个 SQL Server/数据库的安全对象。一个安全对象是一个受保护的资源,只有某些人或过程可以查看或更改,比如一个表中的数据。一个权限使主体获得访问一种特定类型安全对象。

继续护照的比喻,主体是护照的持有人,安全对象是主体想访问的国家,权限是签证可以跨越国界并享有访问。

## 主体(Principals)

主体,在安全上下文是任何用户、组(SQL Server 中称作角色)、或进程中运行的代码,可以请求访问安全对象并且被授予或拒绝访问。所有的 Windows 和 SQL Server 登录名都是主体,以及它们映射到数据库中的用户。下面的列表显示了大多数 SQL Server 中重要主体的层次,从服务器级到 SQL Server 实例级,再到数据库级的主体。

## **Windows 级主体**

->Windows 域登录

->Windows 组

->Windows 本地登录

## **SQL Server 级主体**

->SQL Server 登录

->SQL Server 登录映射到证书

->SQL Server 登录映射到 Windows 登录

->SQL Server 登录映射到非对称密钥

## **数据库级主体**

->应用程序角色

->数据库角色

->数据库用户

->数据库用户映射到证书

->数据库用户映射到 Windows 登录

->数据库用户映射到非对称密钥

->Public 角色

重要的是要理解这个层次,因为一个主体的范围决定了授予它的权限的范围。

例如,数据库用户只有在授予权限的数据库中有权限。一个 SQL Server 级别的主体可以在 SQL Server 上有权限,而 Windows 级别的主体有权限超出 SQL Server 的限制,在 Windows 的本地实例和整个网络。

一个主体可以是一个登录(或用户)或是角色。在 SQL Server 角色的作用类似

于 Windows 组。角色中的成员继承分配给角色的权限。角色使安全管理更容易，因为你不需要为单独的用户管理复杂的权限。SQL Server 支持以下类型的角色：

->固定服务器角色：SQL Server 内置角色执行服务器级任务

->用户定义的服务器角色：你创建的自定义服务器角色，分配服务器级权限，并指定登录名，以便登录名继承服务器对象的权限

->固定数据库角色：内置角色执行数据库任务和分配基本权限

->用户定义的数据库角色：你创建的自定义数据库角色，分配权限，然后添加用户，以使用户继承数据库对象的权限

你可以将用户分配给多个角色。角色也可以嵌套，但是如果你的嵌套太复杂，你会受到性能损失，并且它可能会成为维护和诊断的噩梦。

### **固定服务器角色**

固定服务器角色是 SQL Server 内置的角色，你不能以任何方式改变他们，你只可以给他们添加登录名。它们存在于服务器级别仅用于执行管理任务。SQL Server 中固定服务器角色如下(括号中是实际角色名称)：

->系统管理员(sysadmin)：在 SQL Server 实例执行任何活动。这个角色包含了所有的其他角色，一旦用户是 sysadmin 的成员，他们不需要任何其他角色。sysadmin 的成员可以做任何事情，所以很有必要限制成员用户，只给那些需要并且可以信任的用户访问

->大容量插入管理员(bulkadmin)：执行 BULK INSERT 语句让数据快速导入到数据库中

->创建数据库(dbcreator)：创建和更改数据库

->磁盘管理员(diskadmin)：管理存储数据库的磁盘文件

->进程管理员(processadmin)：管理在 SQL Server 上运行的进程

->服务器管理员(serveradmin)：配置服务器范围的设置。尽管与系统管理员的名称相似，serveradmin 是一个非常不同的和非常有限的角色

->设置管理员(setupadmin)：安装复制和管理扩展程序

->安全管理员(securityadmin)：管理服务器的登录名

固定服务器角色提供的灵活性和安全性允许你将服务器的任务划分成部分。换句话说，如果他们只需要创建数据库，你就不必让他们成为系统管理员的成员。让他们成为 dbcreator 成员，就有他们需要的所有权限。

你可以使用 Management Studio 或 T-SQL 来指定一个登录名到一个固定服务器角色。要使用 Management Studio，执行以下步骤：

- 1、展开，SSMS>对象资源管理器>安全性>登录名
- 2、右键单击 Topaz 登录名并从弹出菜单中选择“属性”
- 3、在“登录属性”对话框中，选择“服务器角色”。这将列出所有可用的服务器角色。注意 Topaz，像所有的登录名一样，已经是 public 角色的成员
- 4、指定登录到 dbcreator 和 diskadmin 角色。图 3.1 显示 Topaz 登录对话框

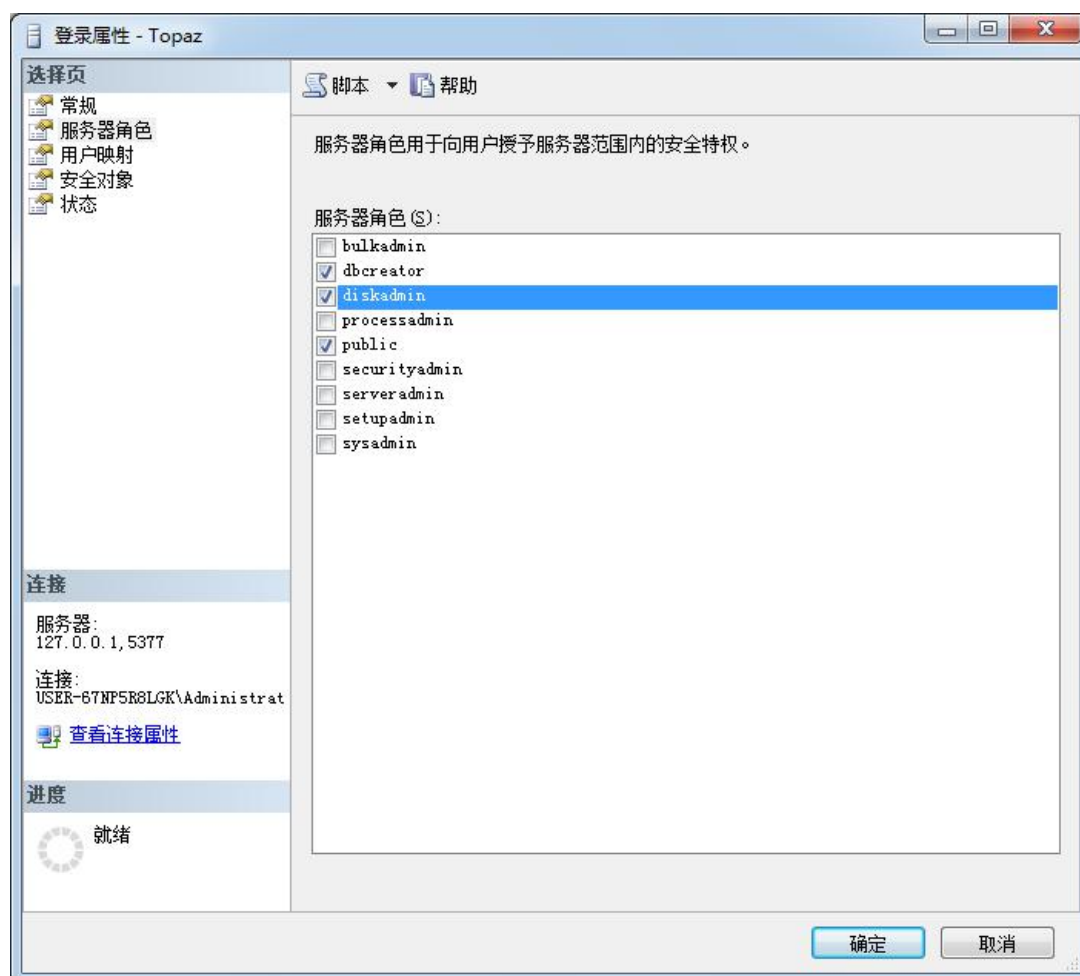


图 3.1 分配 Topaz 登录名到 dbcreator 和 diskadmin 固定服务器角色

#### 5、单击“确定”以保存更改

或者，你也可以通过对象资源管理器>安全性>服务器角色，来添加登录名到对应的角色。下面添加 Topaz 到 securityadmin 服务器角色：

- 1、展开，对象资源管理器>安全性>服务器角色
- 2、右键单击 securityadmin 服务器角色并选择属性，这将打开服务器角色属性窗口
- 3、单击对话框的右下方“添加”按钮，打开“选择服务器登录名或角色”对话框。你可以键入 Topaz，单击检查名称，或单击“浏览”按钮来获得一个登录名列表。一旦你键入 Topaz，对话框如图 3.2 所示

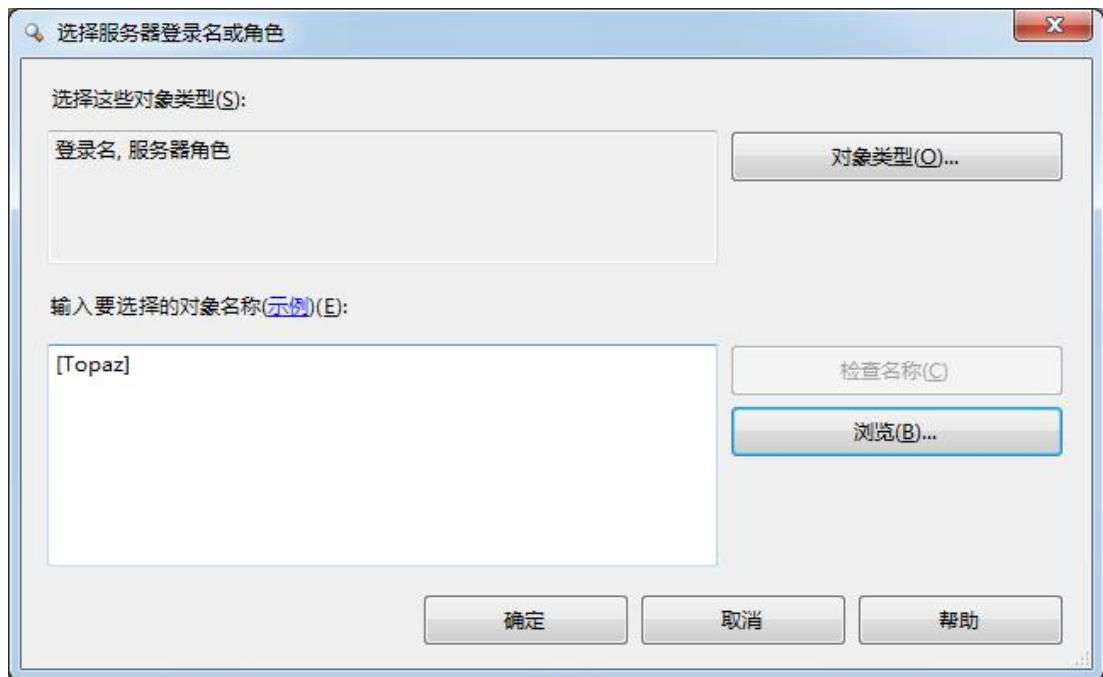


图 3.2 选择 Topaz 添加到服务器角色

4、单击确定添加 Topaz 到服务器角色。服务器角色属性对话框看起来像图

3.3

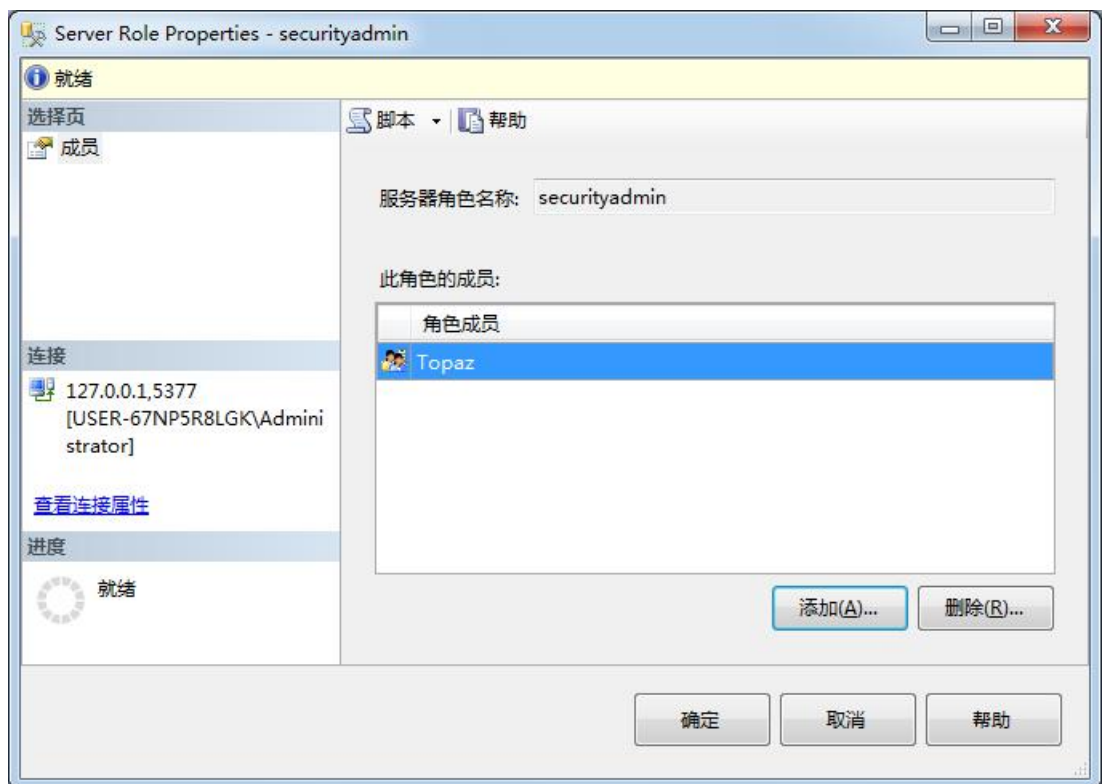


图 3.3 添加 Topaz 到 securityadmin 服务器角色

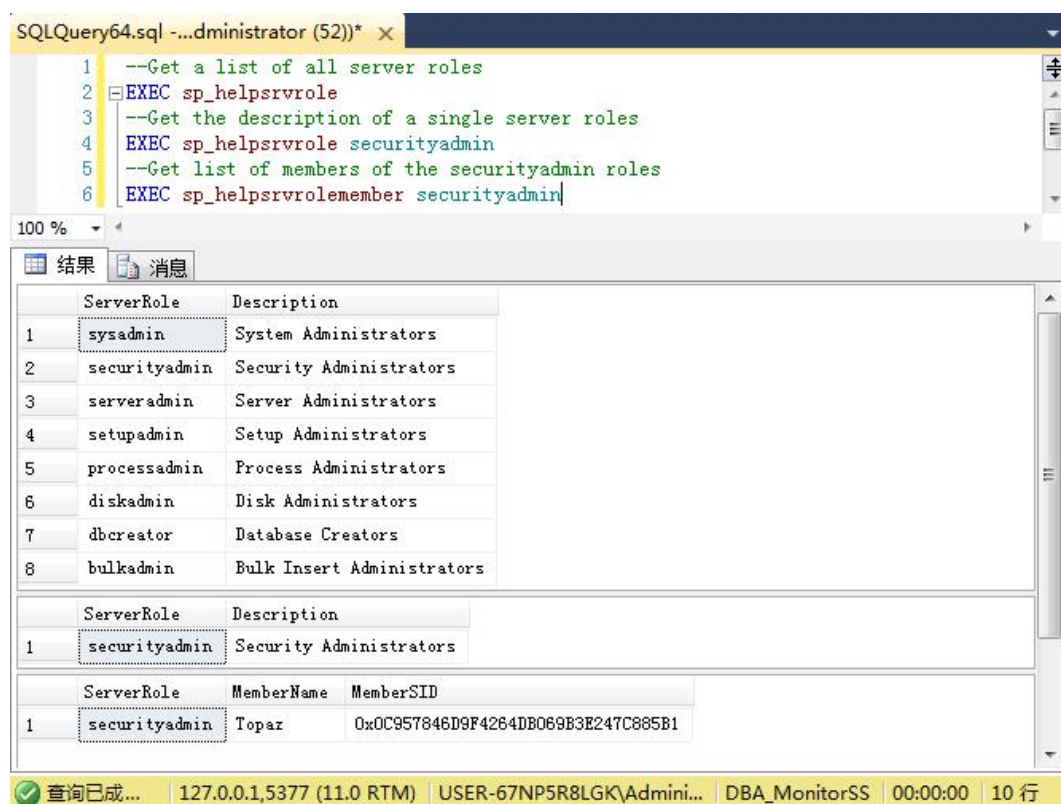
## 5、单击“确定”以保存更改

另一种添加一个登录名到服务器角色的方式是 T-SQL，利用 `sp_addsrvrolemember` 系统存储过程。下面的代码添加到现有的已有的登录名 Topaz 到 sysadmin 角色：

```
EXEC sp_addsrvrolemember 'Topaz', 'sysadmin';
```

### 代码 3.1 添加登录名到服务器角色

你可以通过运行 `sp_helpsrvrole` 和 `sp_helpsrvrolemember` 存储过程找到有关固定服务器角色信息。如果你给 `sp_helpsrvrole` 传入一个有效的服务器角色名称，它将显示该角色的描述；否则显示所有服务器角色。图 3.4 显示了这两个过程的使用方式及返回效果。



The screenshot shows a SQL query window with the following code:

```
1 --Get a list of all server roles
2 EXEC sp_helpsrvrole
3 --Get the description of a single server roles
4 EXEC sp_helpsrvrole securityadmin
5 --Get list of members of the securityadmin roles
6 EXEC sp_helpsrvrolemember securityadmin
```

The results pane shows the output of these queries:

ServerRole	Description
1 sysadmin	System Administrators
2 securityadmin	Security Administrators
3 serveradmin	Server Administrators
4 setupadmin	Setup Administrators
5 processadmin	Process Administrators
6 diskadmin	Disk Administrators
7 dbcreator	Database Creators
8 bulkadmin	Bulk Insert Administrators

ServerRole	Description
1 securityadmin	Security Administrators

ServerRole	MemberName	MemberSID
1 securityadmin	Topaz	0x0C957846D9F4264DB069B3E247C885B1

The status bar at the bottom indicates: 查询已成... | 127.0.0.1,5377 (11.0 RTM) | USER-67NP5R8LGK\Admini... | DBA\_MonitorSS | 00:00:00 | 10 行

图 3.4 使用系统存储过程获取服务器角色信息

## 用户定义的服务器角色

SQL Server 2012 中一个期待已久的安全功能是用户定义的服务器角色。SQL Server 很早就有数据库级别的用户定义的数据库角色,但你终于可以得到颗粒与服务器级别权限的自定义服务器角色。

在 SQL Server 的旧版本,授予用户的某些权限只能通过分配他们到内置固定服务器角色,这样通常有太大的权限。让每个用户成为系统管理员是一个可怕的但常见的做法,一个特别的问题,因为你不能限制系统管理员什么。This violates the principal of least privilege in a big way, but was often a practical necessity。SQL Server 2005 之后,使这一切更精细,让你指定任何特定的服务器级别的用户权限,但是缺少将权限放到一个服务器角色里。

SQL Server 2012 解决了这个问题,支持用户定义的服务器角色。使用 CREATE SERVER ROLE 创建一个新的服务器角色:

```
CREATE SERVER ROLE LimitedDBA;
```

### 代码 3.2 创建服务器角色

你可以授予和拒绝任何想要的服务器级别权限。下面的代码授予 CONTROL SERVER 权限给新的角色,然后拒绝部分权限来缩小的服务器角色的成员的权限。这是一个非常灵活的方式授予用户特定权限。

```
USE master;  
  
GO  
  
--Grant the role virtual sysadmin permissions  
GRANT CONTROL SERVER TO LimitedDBA;  
  
--And take some permissions away
```



```

DENY ALTER ANY LOGIN TO LimitedDBA;

DENY ALTER ANY SERVER AUDIT TO LimitedDBA;

--下面两个注释的 permission_name 不存在(08r2)

--DENY ALTER ANY SERVER ROLE TO LimitedDBA;

--DENY CREATE SERVER ROLE TO LimitedDBA;--Covered by ALTER ANY SERVER ROLE

DENY UNSAFE ASSEMBLY TO LimitedDBA;

```

### 代码 3.3 给服务器角色授予/拒绝权限

为了测试角色,代码 3.4 创建一个映射到 Windows 组的登录名,并添加新的登录名到 LimitedDBA 角色。

```

--Create a login for DBAs Windows group

CREATE LOGIN [Marathon\DBAs] FROM WINDOWS;

--Add to the server role

ALTER SERVER ROLE LimitedDBA ADD MEMBER [Marathon\DBAs];

```

### 代码 3.4 创建登录名并添加到服务器角色

代码 3.5 中创建一个 SQL Server 身份验证的登录名 carol,没有分配任何 SQL Server 实例的权限。然后,尝试在 carol 安全上下文执行各种操作,这些操作需要服务器级别的权限:创建另一个登录名、查看系统信息,并创建另一个服务器角色。所有这些操作都失败了,你可以在图 3.5 中看到,因为 carol 主体没有任何权限来执行这些操作。

```

--Create carol login

CREATE LOGIN carol WITH PASSWORD = 'crolPWD123%%';

```

```

EXECUTE AS LOGIN = 'carol';

--Verify user context

PRINT suser_sname();

--Can Carol alter logins?

CREATE LOGIN donkiely WITH PASSWORD = 'G@Sm3aIKU3HA#fW^
MNYA';--No

--Other server-level permissions?

SELECT * FROM sys.dm_exec_cached_plans;--No, requires VI
EW USER STATE

CREATE SERVER ROLE CarolRole;--No

REVERT;

```

代码 3.5 创建登录并测试它是否有特殊权限

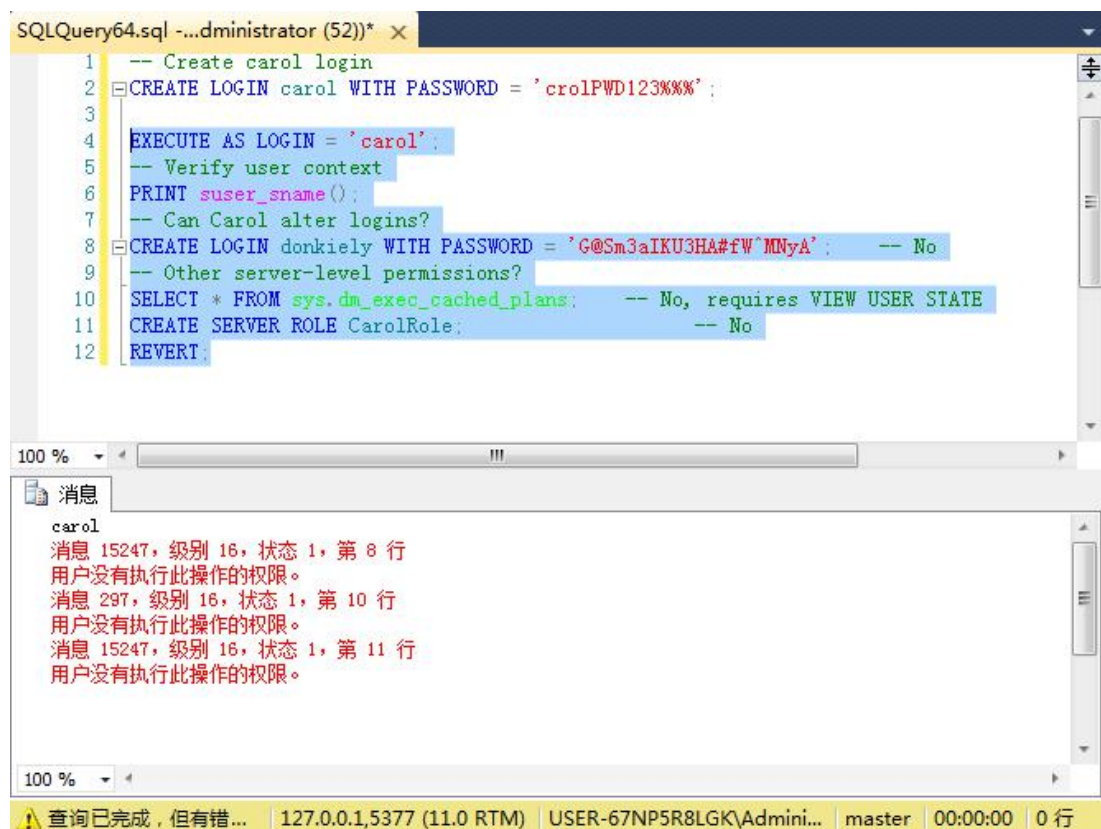


图 3.5 执行失败因为 carol 没有权限

接下来的代码添加 carol 到 LimitedDBA 用户定义服务器角色, 并再次尝试执

行相同的操作。如图 3.6 所示，现在 carol 可以获取系统信息(查询操作)，因为角色授予了 CONTROL SERVER 权限。但 carol 仍然无法创建登录名或服务器角色，因为 LimitedDBA 角色显示拒绝了这些权限。

```
ALTER SERVER ROLE LimitedDBA ADD MEMBER carol;

-- Now does Carol have permissions?

EXECUTE AS LOGIN = 'carol';

CREATE LOGIN donkiely WITH PASSWORD = 'G@Sm3aIKU3HA#fW^MNYA';-- Still not possible

SELECT * FROM sys.dm_exec_cached_plans;-- Yes, CONTROL
SERVER covers VIEW USER STATE

CREATE SERVER ROLE CarolRole;-- Not possible

REVERT;
```

代码 3.6 再次尝试 carol 是否有特殊权限

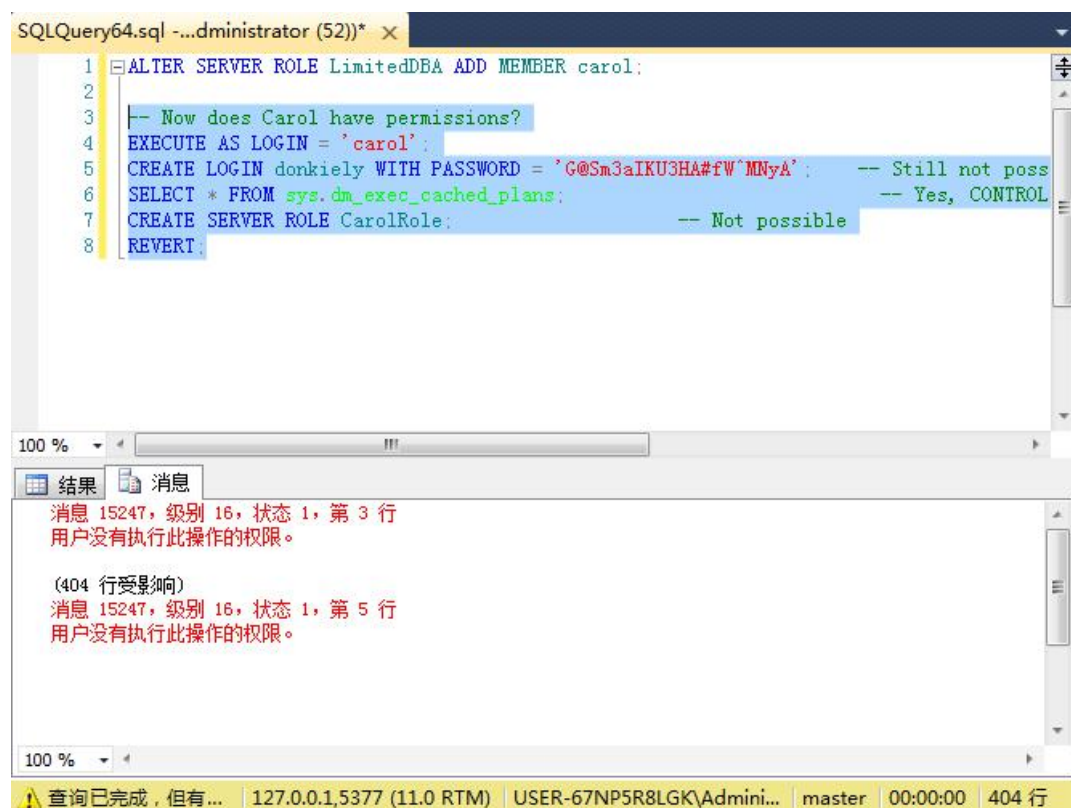
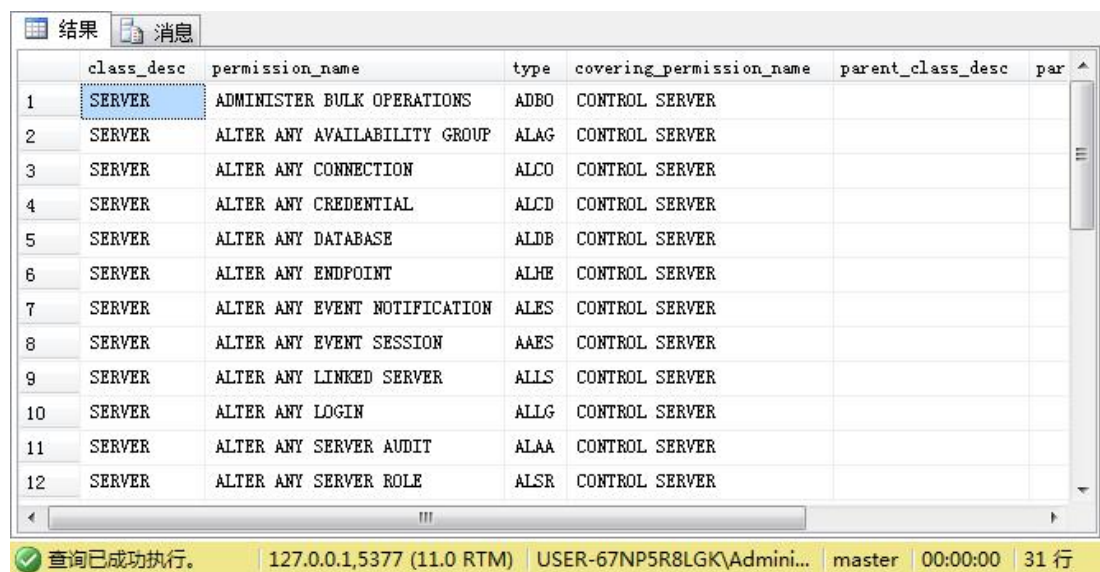


图 3.6 LimitedDBA 角色成员执行结果

为了查看你可以授予和拒绝给服务器角色的所有可用的 server-level 权限，执行以下代码。图 3.7 显示了结果。

```
SELECT * FROM sys.fn_builtin_permissions('SERVER')  
ORDER BY permission_name;
```

代码 3.7 查看所有可用服务器级别权限



	class_desc	permission_name	type	covering_permission_name	parent_class_desc	par
1	SERVER	ADMINISTER BULK OPERATIONS	ADBO	CONTROL SERVER		
2	SERVER	ALTER ANY AVAILABILITY GROUP	ALAG	CONTROL SERVER		
3	SERVER	ALTER ANY CONNECTION	ALCO	CONTROL SERVER		
4	SERVER	ALTER ANY CREDENTIAL	ALCD	CONTROL SERVER		
5	SERVER	ALTER ANY DATABASE	ALDB	CONTROL SERVER		
6	SERVER	ALTER ANY ENDPOINT	ALHE	CONTROL SERVER		
7	SERVER	ALTER ANY EVENT NOTIFICATION	ALES	CONTROL SERVER		
8	SERVER	ALTER ANY EVENT SESSION	AAES	CONTROL SERVER		
9	SERVER	ALTER ANY LINKED SERVER	ALLS	CONTROL SERVER		
10	SERVER	ALTER ANY LOGIN	ALLG	CONTROL SERVER		
11	SERVER	ALTER ANY SERVER AUDIT	ALAA	CONTROL SERVER		
12	SERVER	ALTER ANY SERVER ROLE	ALSR	CONTROL SERVER		

查询已成功执行。 | 127.0.0.1,5377 (11.0 RTM) | USER-67NP5R8LGK\Admini... | master | 00:00:00 | 31 行

图 3.7 服务器级权限的部分列表

你可以创建用户定义的服务器角色，根据用户和组工作需求授予特别的权限，这比早期版本的 SQL Server 更灵活，使安全管理在 SQL Server 2012 更容易，更容易的管理必然意味着一个更安全的服务器。

## 固定数据库角色

固定数据库角色存在于数据库级别，而不是服务器级别，仅在数据库中控制权限。每个数据库都有它自己的固定数据库角色集合，所以你可以单独配置每个数据库中的角色。固定数据库角色类似于固定服务器角色，在这个意义上，它们不能被删除、修改或更改，但可以添加数据库用户和用户定义的角色。固定数据库角色有：

->db\_accessadmin:可以添加或删除数据库中的 Windows 登录名和组和 SQL Server 登录名。

->db\_backupoperator: 可以备份数据库

->db\_datareader:可以查看数据库中所有用户表的数据

->db\_datawriter: 可以增加、修改或删除数据库中所有用户表的数据

->db\_ddladmin:可以在数据库中添加、修改或删除对象。(DDL 代表数据定义语言，使数据库结构变化的一组 T-SQL 命令)

->db\_denydatareader:无法查看数据库中的任何数据

->db\_denydatawriter:无法更改数据库中的任何数据

->db\_owner:可以执行所有的数据库角色以及维护和配置活动。这个角色包含了所有其他的角色，所以它基本上是这个数据库的管理员

->db\_securityadmin:可以在数据库中管理角色成员、语句和对象权限

固定数据库角色可以简化数据库中的权限分配。例如，假设你希望只允许用户备份某个特定的数据库。你不希望用户能够读取数据，只要备份。你能很容易完成这个让用户成为 db\_backupoperator 和 db\_denydatareader 角色的成员。使用 sp\_helprole 和 sp\_helprolemember 系统存储过程来查看有关数据库角色信息。

### **The Public Role and Guest User**

有一些特别的主体需要注意。你可能不会以任何有意义的方式来使用这些主体，但它们确实会影响安全性，所以你需要知道他们是什么。

Public 角色是一种特殊的服务器角色，不能被删除。每个数据库用户都属于这个 Public 角色，所以你不需要为它分配用户、组或角色。每一个 SQL Server 数据库中包含 Public 角色，包括 master、msdb、tempdb 和 model。但是，你可以

授予或限制 Public 角色的权限 根据你的安全需要决定。重要的事情要记住的是，你授予给 Public 角色的权限，适用于所有的数据库用户。

Guest 用户在每个数据库中都存在，包括系统数据库。作为用户，它继承了 Public 角色的权限。它是当一个登录名没有在数据库中映射用户时使用的。默认情况下，Guest 用户没有权限，但你可以授予访问数据库对象的权限，并在数据库中执行操作。正如你所预期的，这是一个非常危险的事情，在一个良好设计的数据库服务器是很少需要的，你应该避免分配权限给 Guest 用户。虽然你不能删除 Guest 用户，你应该在用户数据库中禁用它。

```
USE Northwind;  
  
GO  
  
REVOKE CONNECT FROM guest;  
  
GO
```

### 代码 3.8 通过回收连接权限禁用 Guest 用户

注意：不要禁用系统数据库中的 Guest 用户，这可能会导致你不想处理的问题！这些数据库需要 Guest 用户的各种功能。

### dbo 用户和架构

dbo 是一个特殊的用户帐户在每个数据库映射为 sysadmin 固定服务器角色。这意味着如果你是 sysadmin 角色的成员，你创建的任何数据库对象，对象的所有者会是 dbo，而不是你。你不能删除 dbo 用户，它的映射只有 sysadmin，而不是数据库所有者(db\_owner)。

每个数据库有 dbo 架构，并且是 dbo 用户的默认架构。因此，当你以 sysadmin 访问一个数据库，并创建一个对象，而不指定架构，它的两部分名称将是

dbo.objectname。其他用户访问数据时如果没有指定架构名称，dbo 架构就是它们的第二个默认架构。如果用户 Joe 试图访问表 sales，SQL Server 将首先检查是否有 Joe 默认架构的 sales 表，如果没有，它会检查是否有 dbo.sales。只有在两种架构下都没有找到 sales 表，才会返回错误信息。最佳实践是为每个对象指定一个架构名称。

## **用户定义数据库角色**

数据库角色并不局限于预定义的角色，你可以创建自己的角色。用户可以定义类型的数据库角色：

->标准角色：使用此角色来简化用户组的权限。你可以嵌套固定数据库角色或其他用户定义的角色，并将用户分配给角色，在这种情况下，他们继承了角色的权限。

->应用程序角色：应用程序使用这个角色允许应用程序或连接登录到数据库，并通过提供角色名称和密码来激活应用程序角色。你不能像其他角色添加用户的方式将用户添加到应用程序角色，而且一旦激活，应用程序权限的生命周期存在于连接期间。用户任何单独的权限可能置疑，直到检查应用程序权限。

你可以将用户定义的数据库角色添加到固定数据库角色，类似于将用户添加到固定数据库角色，通过固定数据库角色的属性窗口。

## **安全对象**

一个安全对象是一个你可以访问控制的受保护资源。通常它是一个物理的东西，或者至少是一个可以物理化的数字对象。但安全对象也可以是一个操作，可以对数据库或数据库实例做出某些改变。例如，一个管理员可以授予一个主体拥有某个对象。授予此权限并不能立即更改该对象的所有者，它只是给主体在未来

的时间里有能力去做。

图 3.8 显示了在 SQL Server 实例的大部分安全对象。服务器级别的安全对象具有最广泛的范围，including permissions that affect a principal's ability to make changes to all databases。数据库级别范围包括一个特定的数据库中的所有对象，比如那些用于管理用户以及创建加密密钥的对象。架构范围包括架构内的所有对象，基本上是数据库的数据结构，包括表及其数据。数据库可以包含多种架构，并且每种架构都包含一个完整的数据库对象集合的子集。架构功能的强大之处，你可以指定和拒绝架构权限，并且这些权限适用于架构包含的所有对象。

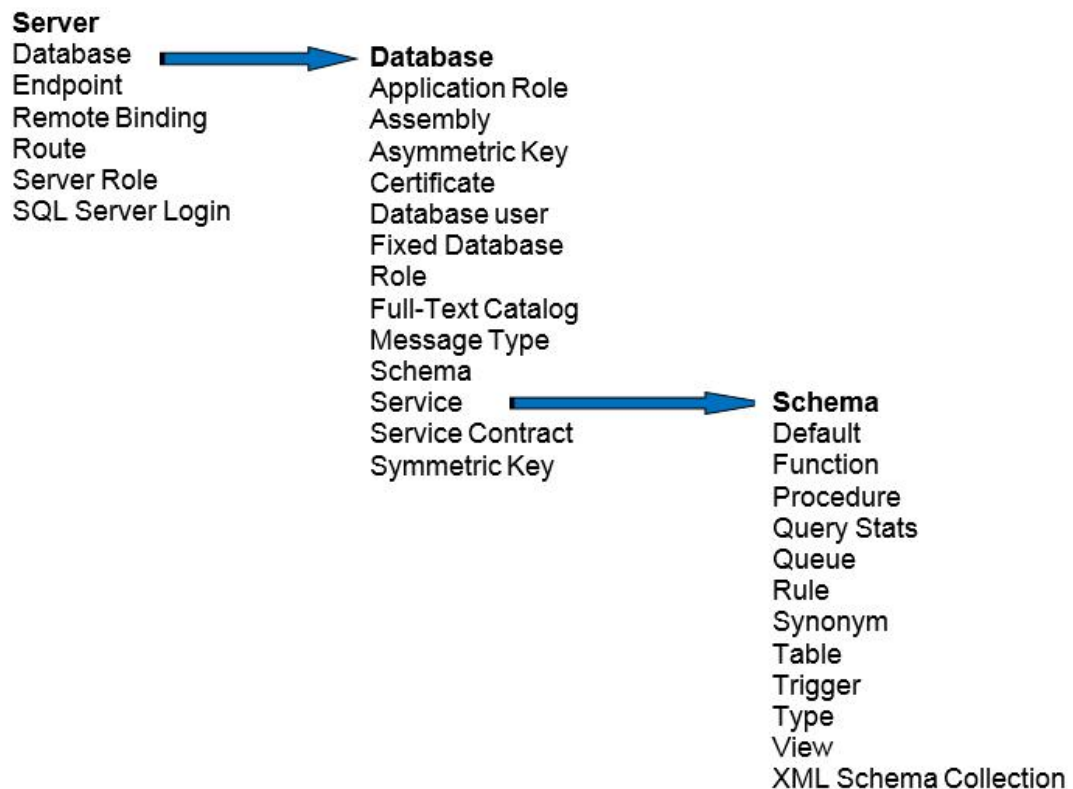


图 3.8 SQL Server 中的安全对象

注意：授予服务器级别的权限通常会影响到小/低范围的权限。例如，授予数据库级权限可能意味着主体对对象在一个或全部数据库架构中具有隐含权限。