SQL Server 安全权限

权限授予主体访问对象，以执行某些操作。SQL Server 有大量你可以授予给主体的权限，你甚至可以拒绝或回收权限。这听起来有点复杂，但在这一系列，你将知道 SQL Server 权限是如何工作的，你可以非常精细地在数据库和服务器对象上控制对象创建、数据访问、以及其他类型操作。

权限

权限像一个签证允许你访问外国，通常有一些基本条件。比如，你只有六个月的期限，你被限制在 3/7 的地区旅行。类似的，SQL Server 权限给主体访问数据库对象做一些事或执行操作。权限可以允许主体读取表的数据或表的某一部分，或能够运行一段特定的代码。甚至允许主体给其他登录名授予权限。你可以给不同的主体授予数百种不同的权限。

在授予权限时，你要遵循最小特权原则。最小特权意味着，你给一个主体需要完成一个任务的权限——没有多也没有少。坚持最小特权原则，是数据库安全的重要步骤。如果主体在数据库中唯一能做的事情是读取产品信息，这个主体就不应该能有意或无意删除表中的内容。本质上是建立一个严密的容器限制主体可以做什么。

权限类型

SQL Server 有许多种你可以授予主体控制访问安全对象的权限。下面列出最常见的权限：

控制 :赋予安全对象的所有可能的权限，使主体成为安全对象的虚拟拥有者。这包括将安全对象授予权限给其他主体。

创建：赋予创建一个特定对象的能力，这取决于它被授予的范围。例如，CREATE DATABASE 权限允许主体在 SQL Server 实例创建新的数据库。

更改：赋予更改安全对象属性的权限，除了更改所有者。这个权限包含了同范围的 alter、create、drop 对象的权限(比如，用户对表 A 有更改权限，那么它能够 add/alter/drop 列、create/drop 索引约束等)例如，一个数据库级别的更改权限，包括更改表和架构权限。

删除：允许一个主体删除任何或所有存储在表中的数据。是一个非常危险的权限！

模拟<登录>或者模拟<用户名>：赋予主体模拟另一个登录/用户。通常用于改变存储过程的执行上下文。

插入：允许主体往表中插入新记录

选择：授予主体从一个特定的表读取数据。这是用户需要的最常见的权限，以便他们可以在表上执行查询。

接管所有权：Confers on a principal permission to take ownership of an object. Granting this permission does not immediately transfer ownership. Instead, it allows the principal to take ownership at some future time.

更新：允许主体更新表中的数据。

查看定义：赋予主体权限查看安全对象的定义。这是一个重要的权限，因为结构信息在数据库攻击中是非常有用的。如果没有这个权限，攻击者发现数据库或服务器实例有价值目标的能力将严重受限。

创建和更改权限可以使用 ANY 关键字：CREATE ANY <object type> and ALTER ANY <object type>.这些权限授予创建或更改任何指定类型安全对象。例如，在数据库级别上授予 ALTER ANY SCHEMA，允许改变数据库中任何架构的属性。在服

务器级别上，ALTER ANY LOGIN 允许主体改变服务器上任何登录名。使用 ANY 关键字，可以灵活的让主体创建或修改一整类的对象，而不是一个单一的对象。只是要注意，在创建和修改权限之间有一些细微的差别。

当你考虑在 SQL Server 安全对象的数量和一个对象可以拥有潜在权限类型的数量时，你开始意识到有颗粒的权限。你可以应用最小特权原则来实现任何用户或角色的权限集，通过给予用户恰当的权限来完成一项任务，而不需要将其他对象暴露。

提示：如果你使用 SQL Server 早期版本，你会意识到，SQL Server 2005 及之后版本完全修改了可用的权限。你不必再将一个用户分配给一个可能有几十个不必要的权限的角色，从而违反了最小特权，并将数据库暴露在有意或偶然的滥用。

这里一个主要考虑是，授予一个权限不一定有效地授予执行一项操作的能力。有时还需要其他权限，提供一个全面的安全上下文来执行敏感操作的能力。一个很常见的例子是创建表权限。授予这个权限，理论上允许一个主体在特定的数据库下创建表。但主体还需要有能力在一个架构中创建表。如果主体没有对任何架构的更改权限，它将无法创建一个表。

权限语句

即使你是通过图形化工具分配权限，底层还是执行 TSQL 权限语句。

GRANT：授予对安全对象或操作的权限给主体

REVOKE："Un-grant"权限，取消一个早期做的 grant 语句。撤销的权限仍然可以通过在具有权限的组或角色中继承。当你创建新的对象时，revoke 是默认的权限状态，所以特定权限没有授予，但是可以继承。

DENY：拒绝撤销权限，使它不能被继承。这是最具限制性的权限，并且优

先于所有其他权限。DENY 不适用于 sysadmin 角色的成员或对象的所有者。

不要低估 DENY 权限的重要性。例如，假设你有一个临时雇员，他是进来做数据输入，你不希望他能够编辑或删除现有记录。你已经分配给 editor 角色(公司内部人员都属于)对某些表的足够的权限。你可以创建一个特殊的登录名，然后在适当的表上 deny UPDATE 和 DELETE 权限。临时雇员可以从 editor 角色继承足够的权限输入新记录，但是不能修改/删除已存在的记录。

授予权限

在 Management Studio 下授予权限的最灵活的方法是修改数据库用户或角色的属性。你还可以通过修改单个对象的属性来授予权限，但这种方法不灵活，并且具有更高的维护成本。

下面的练习将在 AdventureWorks2012 数据库创建自定义数据库角色。这个角色的成员，需要必要的权限插入和更新一些 HR 相关的表，并且能够执行相关的存储过程，但没有其他的特殊权限。

提示：一旦你为一个用户配置权限，然后需要为另一个用户重复上述过程，授予和第一个用户同样的权限，那么把权限分配给一个角色，然后简单地将用户添加到角色即可。

你可以利用第二篇在 AdventureWorks2012 数据库下创建的用户 Topaz。如果你没有创建用户，执行代码 4.1 创建登录名和用户。

```
    USE master;
    GO
    CREATE LOGIN Topaz WITH PASSWORD = 'yBqyZIPT8}b]b[{5al0
v';
    GO
```

```
USE AdventureWorks2012;
GO
CREATE USER Topaz FOR LOGIN Topaz
    WITH DEFAULT_SCHEMA = HumanResources;
GO
```

代码 4.1 创建创建名并映射到数据库用户

在 AdventureWorks2012 数据库创建一个 DataEntry 自定义数据库角色，参考如下步骤：

1、SSMS 连接到安装有 AdventureWorks2012 数据库的实例。对象资源管理器->数据库->AdventureWorks2012->安全性->角色->数据库角色

2、右击数据库角色，弹出菜单选择新建数据库角色

3、角色名称键入 DataEntry，所有者 dbo

4、点击添加按钮，将用户 Topaz 添加到角色(如果用户不存在请先创建)。在你关闭选择数据库用户或角色对话框后，数据库角色-新建对话框应该如图 4.1 所示

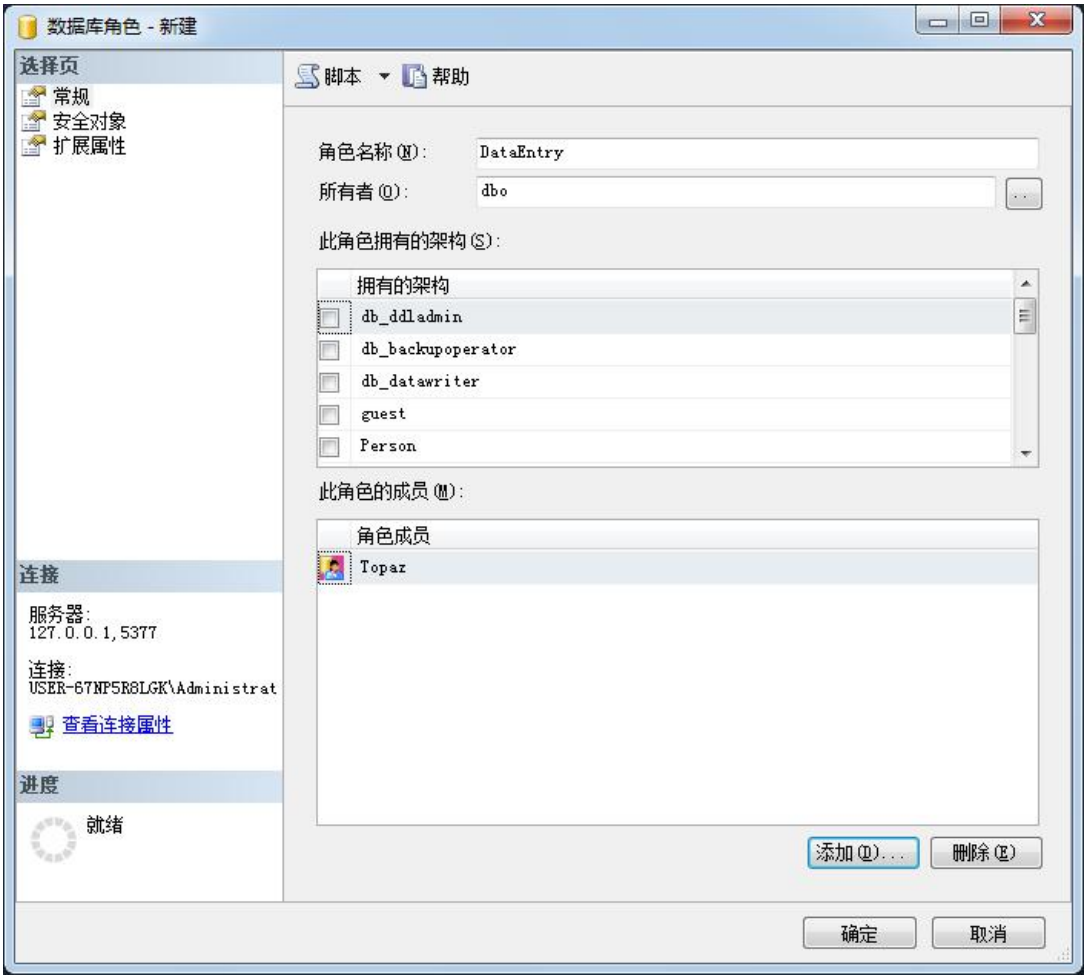


图 4.1 创建 DataEntry 数据库角色并添加 Topaz 用户

5、点击确定保存修改并创建角色

当前 DataEntry 角色并不允许角色中的成员能做什么，因为你还没有给角色分配任何权限。DataEntry 角色中的成员需要能够往表 Employee、Address、JobCandidate 插入和更新数据，同时他们需要执行 uspUpdateEmployeeHireInfo 和 uspUpdateEmployeePersonalInfo 存储过程。但是他们不能查看存储过程的定义。

使用下面的步骤给 DataEntry 角色添加合适的权限

1、对象资源管理器->数据库->AdventureWorks2012->安全性->角色->数据库角色->DataEntry

2、右击 DataEntry，弹出菜单选择属性。打开数据库角色属性对话框

3、左侧选择安全对象，这个页面让你选择角色有权操作的安全对象，并指定对安全对象上的权限

4、点击“搜索”按钮添加安全对象。这将打开“添加对象”对话框，该对话框提供了特定对象、特定类型的所有对象、属于该架构的所有对象的选项。在本例中，因为你想为表和存储过程中添加权限，图 4.2 中保持默认选择-特定对象，然后单击“确定”

图 4.2 选择添加对象

5、打开“选择对象”对话框。单击“对象类型”按钮打开“选择对象类型”对话框，然后从列表中选择“存储过程”和“表”，如图 4.3 所示。请单击“确定”以关闭该对话框并返回“选择对象”对话框，该对话框现在看起来像图 4.4。你将看到在对象类型框中列出的存储过程和表

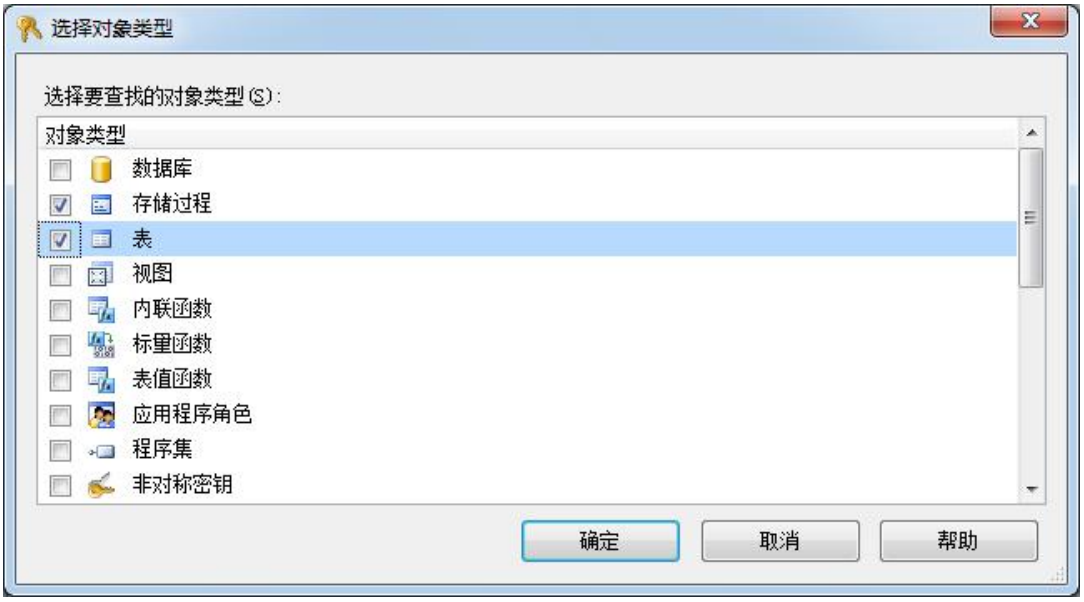


图 4.3 选择对象类型

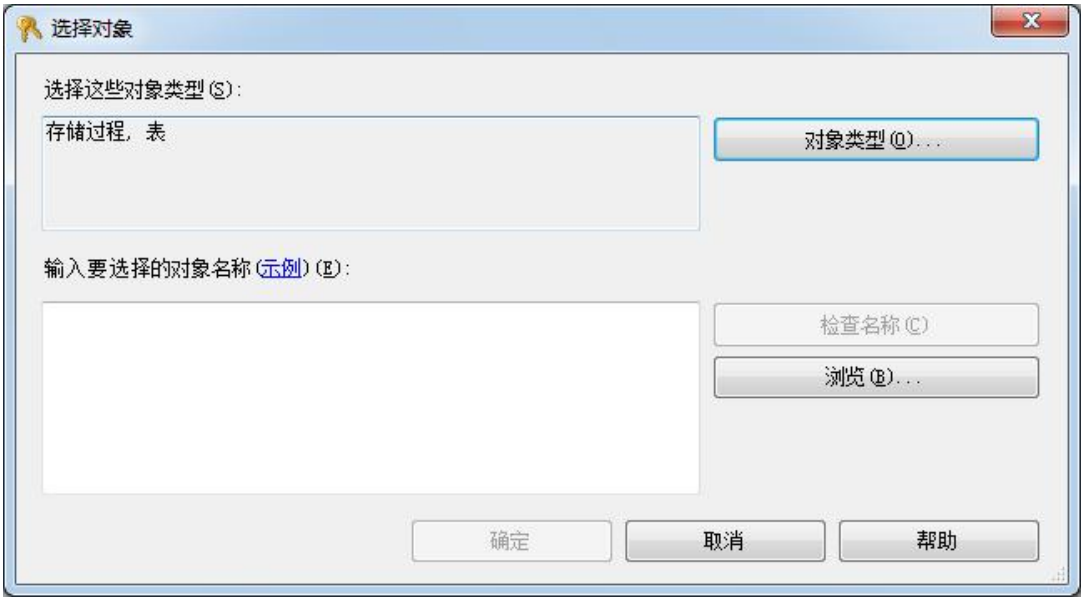


图 4.4 选择对象

6、单击“浏览”按钮以查看数据库中的存储过程和表的列表。这将打开“查找对象”对话框，向下滚动查找并选择对象。

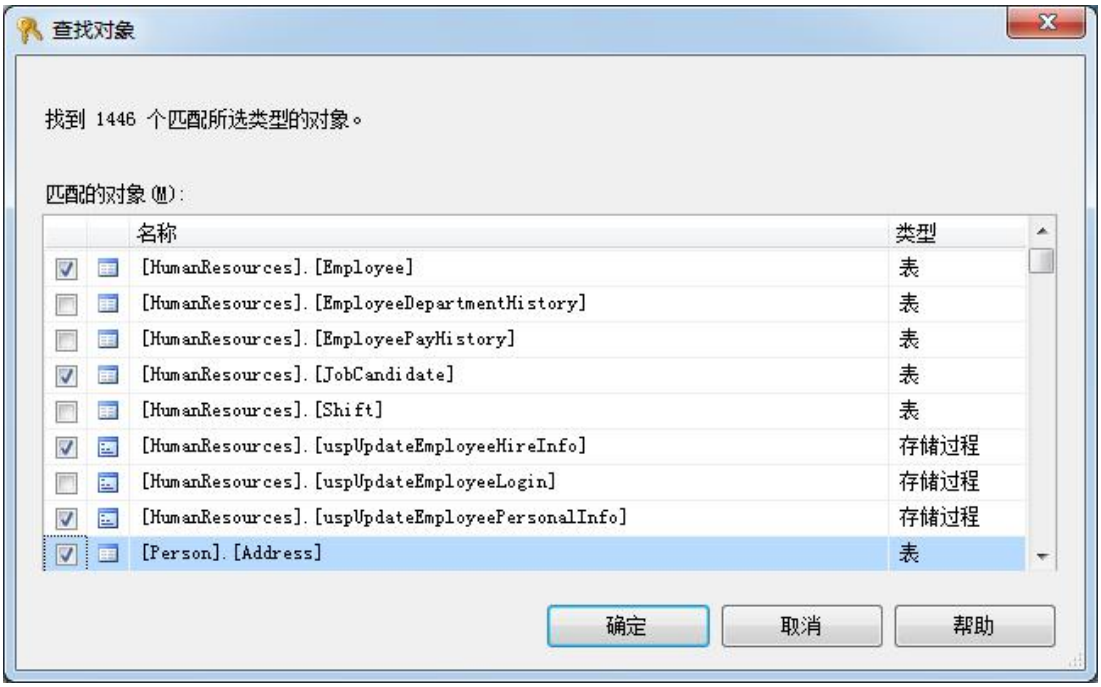


图 4.5 选择存储过程和表

7、单击“确定”以关闭“查找对象”对话框中。此时你选择的对象以分号分隔列在选择对象对话框中，如图 4.6 所示。请单击“确定”以关闭该对话框并保存更改

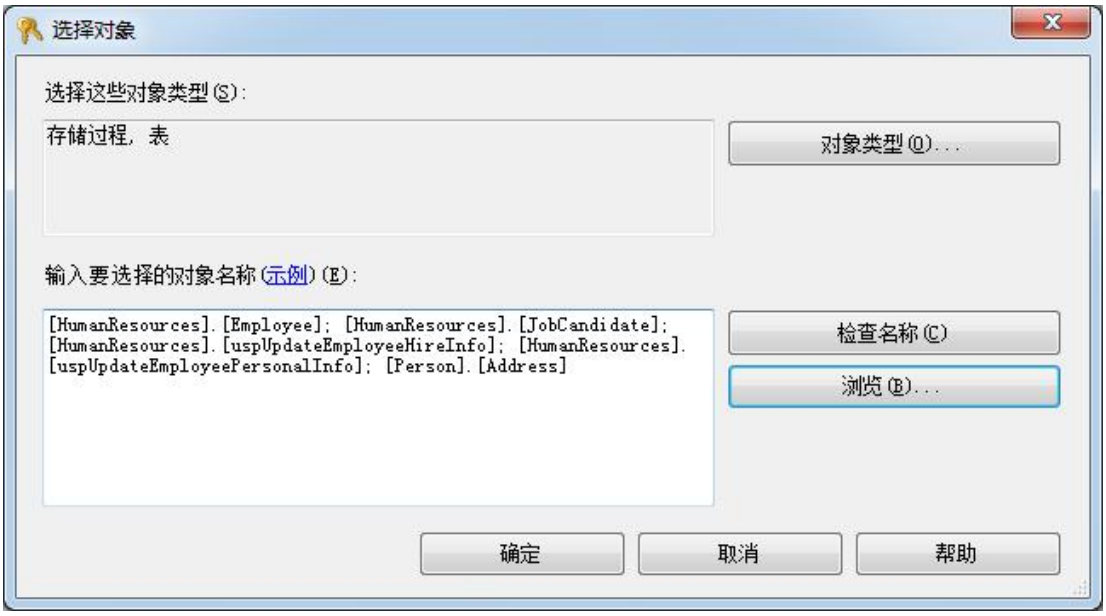


图 4.6 选择对象结果

8、现在 DataEntry 数据库角色属性对话框列出你选择的安全对象，并且列出

每个对象可用的权限。DataEntry 角色成员需要可以插入和更新数据到这些表，逐一选择表，在对话框的下部勾选插入/更新授予列的复选框。图 4.7 显示了 HumanResources.Employee 表的权限

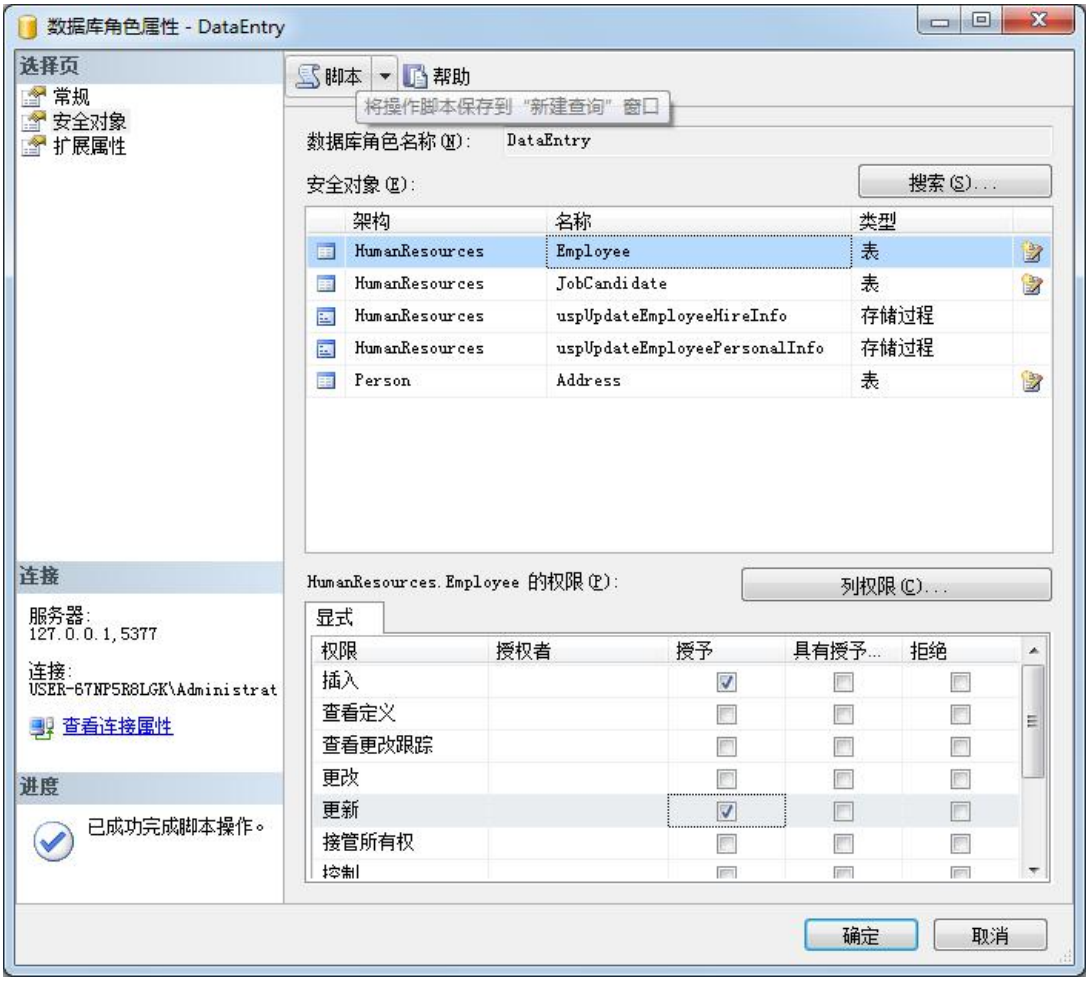


图 4.7 在表上授予插入/更新权限

提示：授予(Grant)允许指定的权限，具有授予权限(With Grant)允许用户或角色授予其他主体的权限。小心 With Grant 权限！

9、对于每一个存储过程，授予执行权限并拒绝查看定义权限。图 4.8 显示了 useUpdateEmployeeHireInfo 存储过程的权限

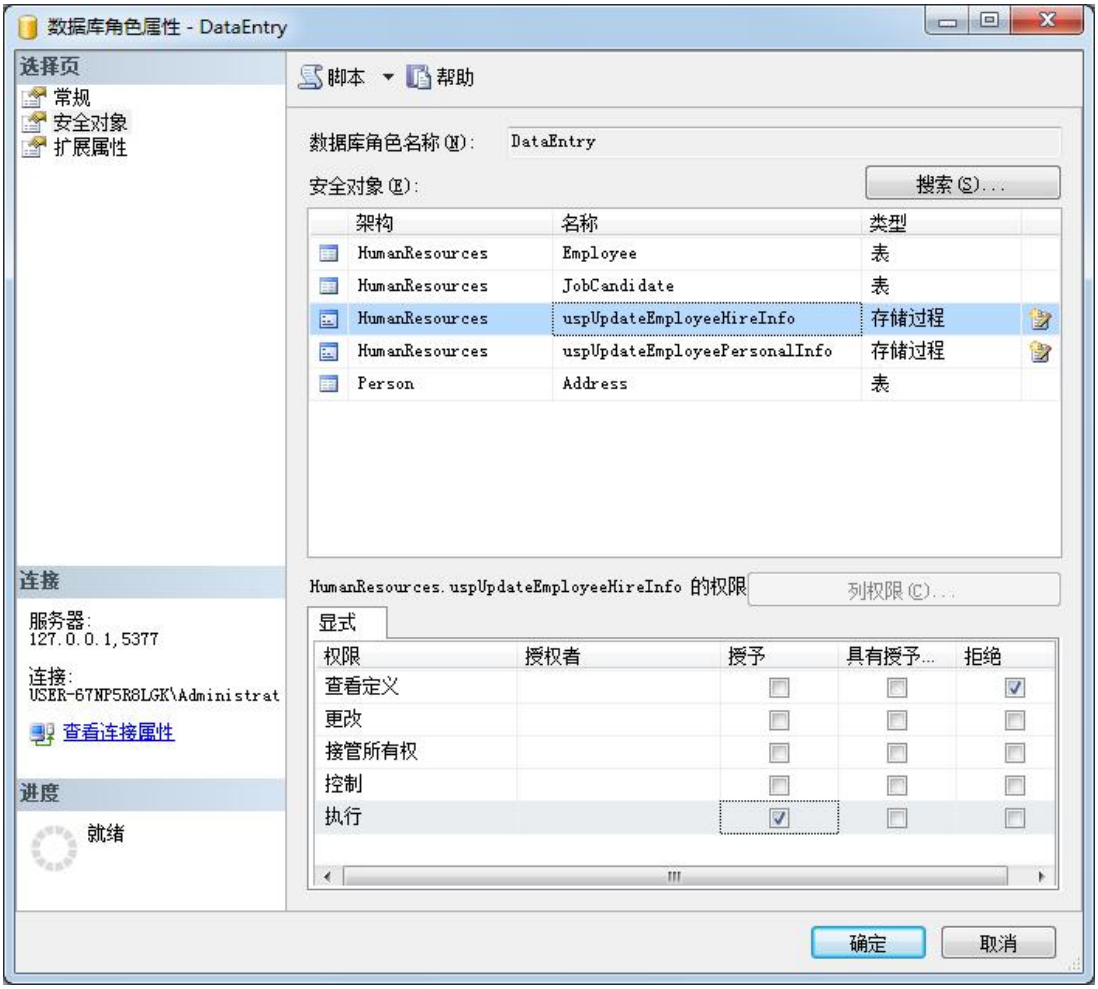


图 4.8 授予执行过程权限、拒绝查看定义权限

10、在“数据库角色属性”对话框中单击“确定”以保存你的更改并提交到数据库。根据选定的对象和权限的数量，这可能需要一些时间。

当然，你可以使用 T-SQL 代码创建对象并分配权限。代码 4.2 创建 DataEntry 角色，添加 Topaz 用户，然后分配和图形界面相同的权限。

```
    -- Create the DataEntry role and assign Topaz to it
    CREATE ROLE [DataEntry] AUTHORIZATION [dbo];
    ALTER ROLE [DataEntry] ADD MEMBER [Topaz];
    -- Assign permissions to the DataEntry role
    GRANT INSERT ON [HumanResources].[Employee] TO [DataEnt
ry];
```

```
    GRANT UPDATE ON [HumanResources].[Employee] TO [DataEnt
ry];

    GRANT INSERT ON [HumanResources].[JobCandidate] TO [Dat
aEntry];

    GRANT UPDATE ON [HumanResources].[JobCandidate] TO [Dat
aEntry];

    GRANT INSERT ON [Person].[Address] TO [DataEntry];

    GRANT UPDATE ON [Person].[Address] TO [DataEntry];

    GRANT EXECUTE ON [HumanResources].[uspUpdateEmployeeHi
reInfo] TO [DataEntry];

    DENY VIEW DEFINITION ON [HumanResources].[uspUpdateEmpl
oyeeHireInfo] TO [DataEntry];

    GRANT EXECUTE ON [HumanResources].[uspUpdateEmployeePe
rsonalInfo] TO [DataEntry]

    DENY VIEW DEFINITION ON [HumanResources].[uspUpdateEmpl
oyeePersonalInfo] TO [DataEntry];

    GO
```

代码 4.2 语句创建角色并分配权限

检查和测试权限

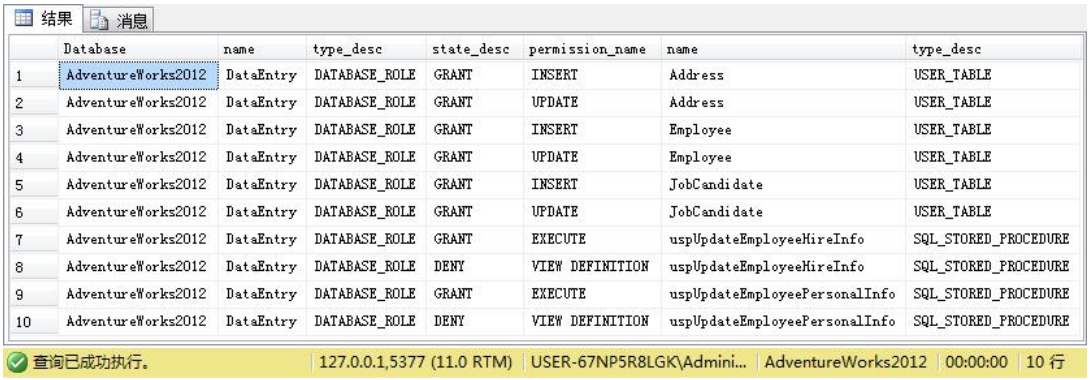
如果你想检查 DataEntry 角色的权限，你可以使用 GUI 工具或执行 T-SQL 代码访问数据库对象的元数据。使用 GUI，对象资源管理器->数据库->AdventureWorks2012->安全性->角色->数据库角色->DataEntry->属性。这将打开和创建角色相同的"数据库角色属性"对话框，你可以使用安全对象审查该角色的权限。

或者你可以使用代码 4.3 中的 T-SQL 查看 DataEntry 角色的权限，利用 sys.database_permissions 和 sys.database_principals 安全目录视图和 sys.objects 目录视图。

```
    SELECT DB_NAME() AS 'Database', p.name, p.type_desc, db
p.state_desc,

        dbp.permission_name, so.name, so.type_desc

    FROM sys.database_permissions dbp

        LEFT JOIN sys.objects so ON dbp.major_id = so.object
_id

        LEFT JOIN sys.database_principals p ON dbp.grantee_
principal_id = p.principal_id

    WHERE p.name = 'DataEntry'

    ORDER BY so.name, dbp.permission_name;
```

代码 4.3 查看 DataEntry 角色的权限

当你执行上面代码，你将会看到结果如图 4.9 所示



结果 消息

	Database	name	type_desc	state_desc	permission_name	name	type_desc
1	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	INSERT	Address	USER_TABLE
2	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	UPDATE	Address	USER_TABLE
3	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	INSERT	Employee	USER_TABLE
4	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	UPDATE	Employee	USER_TABLE
5	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	INSERT	JobCandidate	USER_TABLE
6	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	UPDATE	JobCandidate	USER_TABLE
7	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	EXECUTE	uspUpdateEmployeeHireInfo	SQL_STORED_PROCEDURE
8	AdventureWorks2012	DataEntry	DATABASE_ROLE	DENY	VIEW DEFINITION	uspUpdateEmployeeHireInfo	SQL_STORED_PROCEDURE
9	AdventureWorks2012	DataEntry	DATABASE_ROLE	GRANT	EXECUTE	uspUpdateEmployeePersonalInfo	SQL_STORED_PROCEDURE
10	AdventureWorks2012	DataEntry	DATABASE_ROLE	DENY	VIEW DEFINITION	uspUpdateEmployeePersonalInfo	SQL_STORED_PROCEDURE

查询已成功执行。 | 127.0.0.1,5377 (11.0 RTM) | USER-67NP5R8LGK\Admini... | AdventureWorks2012 | 00:00:00 | 10 行

图 4.9 DataEntry 角色的权限

你可以通过打开新建查询->更改连接用 Topaz 登录测试这些权限。然后尝试在 HumanResources.Employee、HumanResources.JobCandidate、Person.Address 表插入新行或更新已有数据，并执行分配权限的存储过程。这些操作应该成功。然后尝试插入或更新其他表中的数据，这些操作应该失败。你应该只能够执行已经授予的权限。

你可以用代码 4.4 执行相同的测试。代码开始设置的执行上下文为 Topaz，DataEntry 角色的成员。然后插入新行到 Person.Address 表。这一操作成功，因为

角色允许在 Person.Address 表插入数据。然后代码试图插入新行到 HumanResources.Department 表，插入失败，因为角色在此表没有 insert 权限。接下来的代码成功执行 HumanResources.uspUpdateEmployeePersonalInfo 存储过程；执行 dbo.uspGetManagerEmployees 失败，因为角色没有此过程执行权限。最后，代码将执行上下文切回到登录时的安全上下文。

```
    EXECUTE AS USER = 'Topaz';
    -- Succeeds
    INSERT INTO Person.Address
        (AddressLine1, City, StateProvinceID, PostalCode)
    VALUES
        ('8 Hazelnut', 'Irvine', 9, '92602');
    GO
    -- Fails
    INSERT INTO HumanResources.Department
        (Name, GroupName)
    VALUES
        ('Advertising', 'Sales and Marketing');
    GO
    -- Succeeds (doesn't actually change any data)
    DECLARE @RC INT;
    EXECUTE @RC = HumanResources.uspUpdateEmployeePersonal
Info
       1, '295847284', '1963-03-02', 'S', 'M';
    GO
    -- Fails
    EXECUTE dbo.uspGetManagerEmployees 1;
```

```
GO

REVERT;
```

代码 4.4 测试 DataEntry 角色的权限