**Steps to Securing your SQL Server**

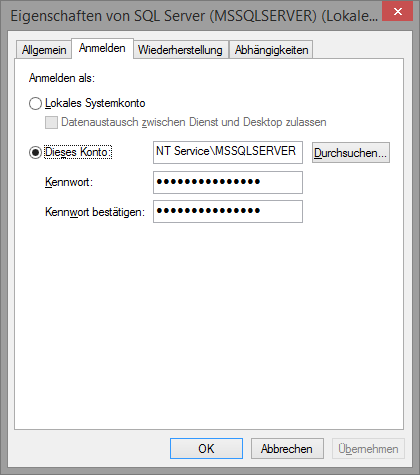
### Führen sie alle Tests mit der Kino-Anwendung durch:

### Registry: Mit SQL-Server Authentifizierung:

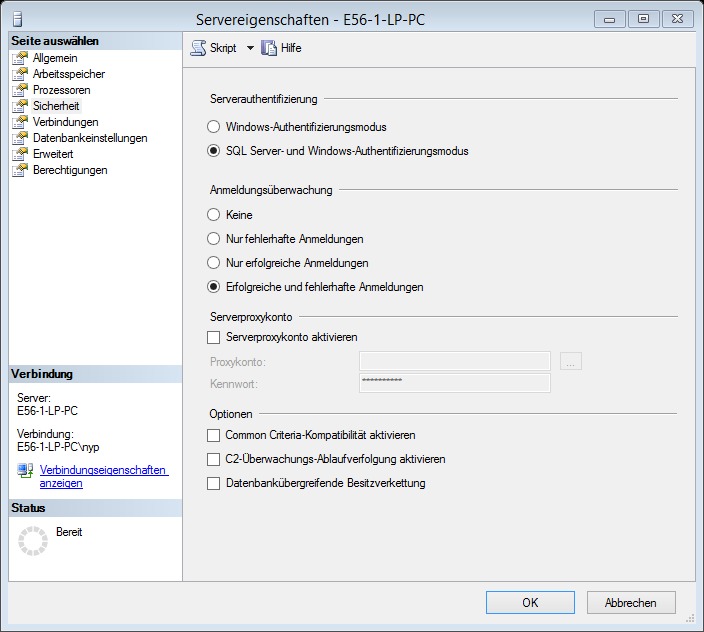
### Registry: Mit Windows Authentifizierung:

### Diese Arbeit ist abzugeben

### Change the Startup Account (tun sie es und dokumentieren sie es hier)

* Erstellen sie einen Account zum Starten des SQL-Servers.
* Testen sie: Datenbankerstellen, Backup/Restore, Kinoanwendung  
  

### Auditing of Failed Logins and Denied Access (tun sie es zu Hause und dokumentieren sie es hier)

* Turn on the Profiler!
* Add a registry key!
* Zeigen sie hier einen Screenshot ihrer Logdatei: gute Zugriffe, schlechte Zugriffe! 

### Protect Your Extended Stored Procedures (tun sie es und dokumentieren sie es hier)

* In welcher Datenbank befindet sich ***sys.xp\_cmdshell***?
* Schauen sie sich diesen Artikel an: <http://www.aspfaq.com/sql2005/show.asp?id=29>
* Erteilen sie die Berechtigung zum Ausführen der ***sys.xp\_cmdshell!***
* Was kann ***cmdshell***? Geben sie mindestens 3 Beispiele!

### Change the SQL Server Port and Block it

### 

**Hauptaufgabe 1**

Richten sie zwei Accounts(SQL-Server Authentifizierung) ein:

Account ‚***SKinoAdministrator’***

Account ‚ ***SKinoKasse‘***

Dabei soll so wenig Zugriff wie möglich gewährt werden:

***SKinoKasse:*** kann Reservierungen durchführen, kann keine Stammdaten bearbeiten

***SKinoAdministrator:*** kann Stammdaten bearbeiten, kann keine Reservierungen durchführen

Sichern sie ihre SQLs für ihre Dokumentation!

USE [master]

GO

CREATE LOGIN [SKinoAdministrator]

WITH PASSWORD='sql' MUST\_CHANGE,

DEFAULT\_DATABASE=[master],

CHECK\_EXPIRATION=ON, CHECK\_POLICY=ON

GO

USE [KinoSQL]

GO

CREATE USER [SKinoAdministrator]

FOR LOGIN [SKinoAdministrator]

GO

USE [KinoSQL]

GO

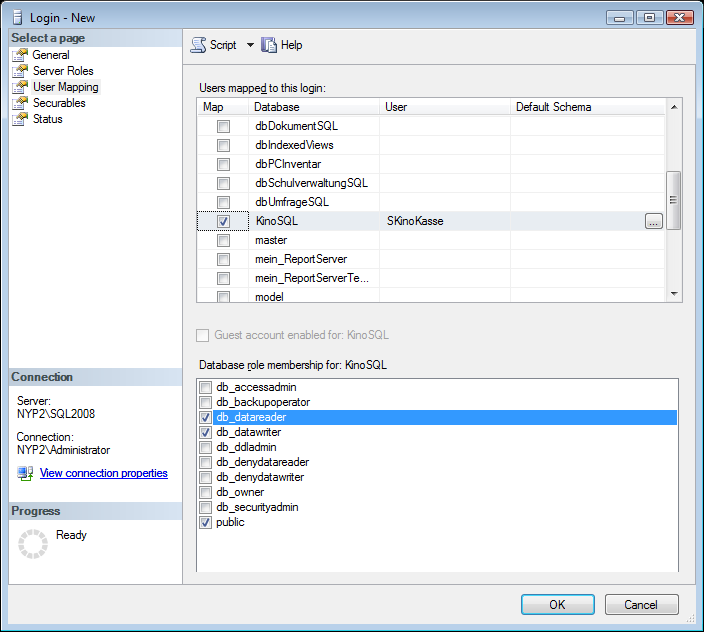
EXEC sp\_addrolemember 'db\_datareader', 'SKinoAdministrator'

GO

USE [KinoSQL]

GO

EXEC sp\_addrolemember 'db\_datawriter', 'SKinoAdministrator'



USE [master]

GO

CREATE LOGIN [SKinoKasse] WITH PASSWORD='sql',

DEFAULT\_DATABASE=[KinoSQL], CHECK\_EXPIRATION=OFF, CHECK\_POLICY=OFF

GO

USE [KinoSQL]

GO

CREATE USER [SKinoKasse] FOR LOGIN [SKinoKasse]

GO

USE [KinoSQL]

GO

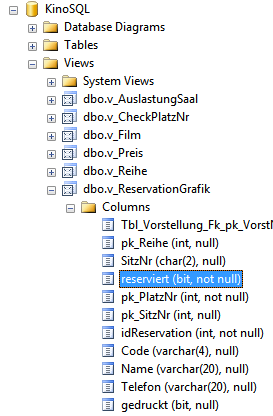
EXEC sp\_addrolemember 'db\_datareader', 'SKinoKasse'

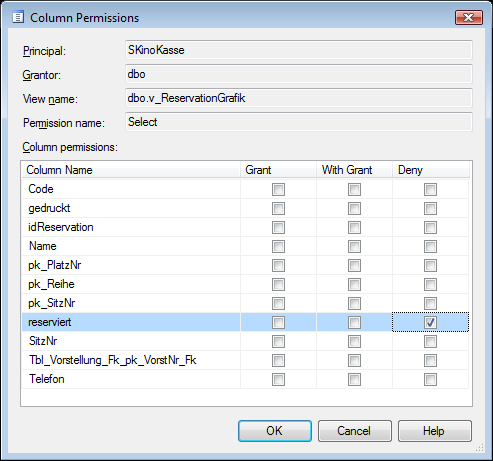
GO

USE [KinoSQL]

GO

EXEC sp\_addrolemember 'db\_datawriter', 'SKinoKasse'



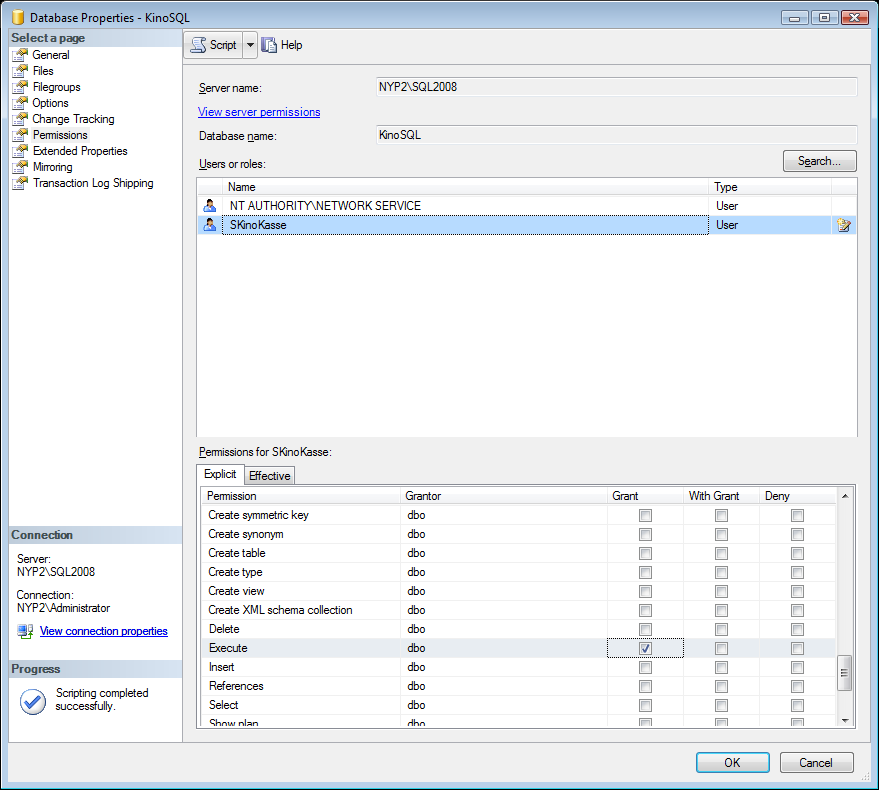


use [KinoSQL]

GO

DENY SELECT ON [dbo].[v\_ReservationGrafik] ([idReservation]) TO [SKinoAdministrator]

Ausführen gespeicherter Prozeduren:



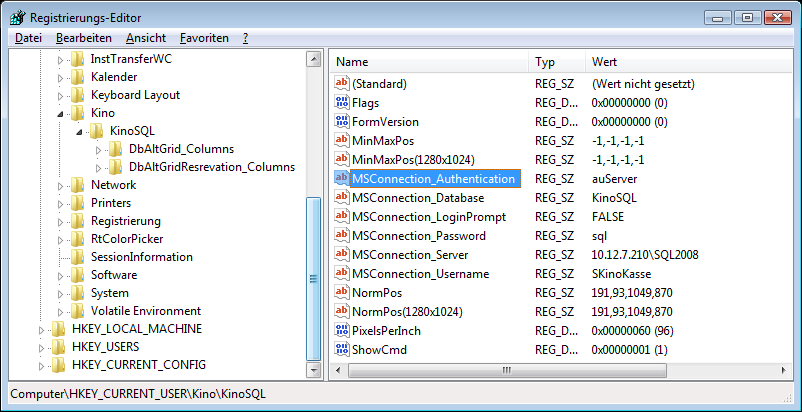
use [KinoSQL]

GO

GRANT EXECUTE TO [SKinoKasse]

GO

Registryeinträge:



**Hauptaufgabe 2**

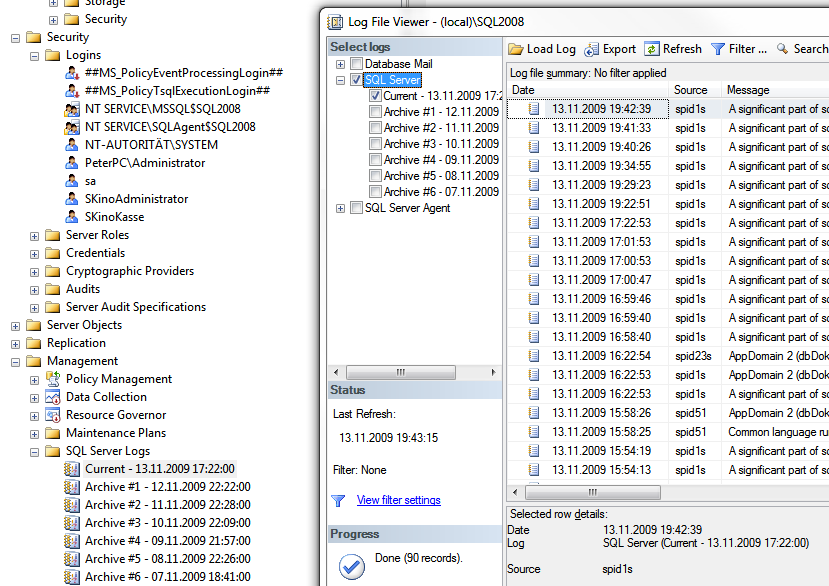
Richten sie zwei Accounts (Windows Authentifizierung) ein:

***SKinoKasse:*** kann Reservierungen durchführen, kann keine Stammdaten bearbeiten

***SKinoAdministrator:*** kann Stammdaten bearbeiten, kann keine Reservierungen durchführen

Sichern sie ihre SQLs für ihre Dokumentation!

Zugriff beweisen mit Log-Datei/Screenshot, etc.!

****

**Reserveaufgabe 1**

Erstellen sie ein Zertifikat mit T-SQL und verschlüsseln Sie einen einfachen String!

Benutzen sie dabei unterschiedliche Verschlüsselungen.

*Tipp*: <http://msdn2.microsoft.com/en-us/library/ms174430.aspx>

*Syntaxtipps*:

CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'password'

==================================================================================================================

CREATE CERTIFICATE certificate\_name [ AUTHORIZATION user\_name ]

{ FROM <existing\_keys> | <generate\_new\_keys> }

[ ACTIVE FOR BEGIN\_DIALOG = { ON | OFF } ]

<existing\_keys> ::=

ASSEMBLY assembly\_name

| {

[ EXECUTABLE ] FILE = 'path\_to\_file'

[ WITH PRIVATE KEY ( <private\_key\_options> ) ]

}

<generate\_new\_keys> ::=

[ ENCRYPTION BY PASSWORD = 'password']

WITH SUBJECT = 'certificate\_subject\_name'

[ , <date\_options> [ ,...n ] ]

<private\_key\_options> ::=

FILE = 'path\_to\_private\_key'

[ , DECRYPTION BY PASSWORD = 'password' ]

[ , ENCRYPTION BY PASSWORD = 'password' ]

<date\_options> ::=

START\_DATE = 'mm/dd/yyyy' | EXPIRY\_DATE = 'mm/dd/yyyy'

==================================================================================================================

CREATE SYMMETRIC KEY key\_name [ AUTHORIZATION owner\_name ]

WITH <key\_options> [ , ... n ]

ENCRYPTION BY <encrypting\_mechanism> [ , ... n ]

<encrypting\_mechanism> ::=

CERTIFICATE certificate\_name

|

PASSWORD = 'password'

|

SYMMETRIC KEY symmetric\_key\_name

|

ASYMMETRIC KEY asym\_key\_name

<key\_options> ::=

KEY\_SOURCE = 'pass\_phrase'

|

ALGORITHM = <algorithm>

|

IDENTITY\_VALUE = 'identity\_phrase'

<algorithm> ::=

DES | TRIPLE\_DES | RC2 | RC4 | RC4\_128

| DESX | AES\_128 | AES\_192 | AES\_256

==================================================================================================================

CREATE ASYMMETRIC KEY Asym\_Key\_Name

[ AUTHORIZATION database\_principal\_name ]

{

FROM <Asym\_Key\_Source>

|

WITH ALGORITHM = { RSA\_512 | RSA\_1024 | RSA\_2048 }

}

[ ENCRYPTION BY PASSWORD = 'password' ]

<Asym\_Key\_Source>::=

FILE = 'path\_to\_strong-name\_file'

|

EXECUTABLE FILE = 'path\_to\_executable\_file'

|

ASSEMBLY Assembly\_Name

==================================================================================================================

OPEN SYMMETRIC KEY Key\_name DECRYPTION BY <decryption\_mechanism>

<decryption\_mechanism> ::=

CERTIFICATE certificate\_name [ WITH PASSWORD = 'password' ]

|

ASYMMETRIC KEY asym\_key\_name [ WITH PASSWORD = 'password' ]

|

SYMMETRIC KEY decrypting\_Key\_name

|

PASSWORD = 'decryption\_password'