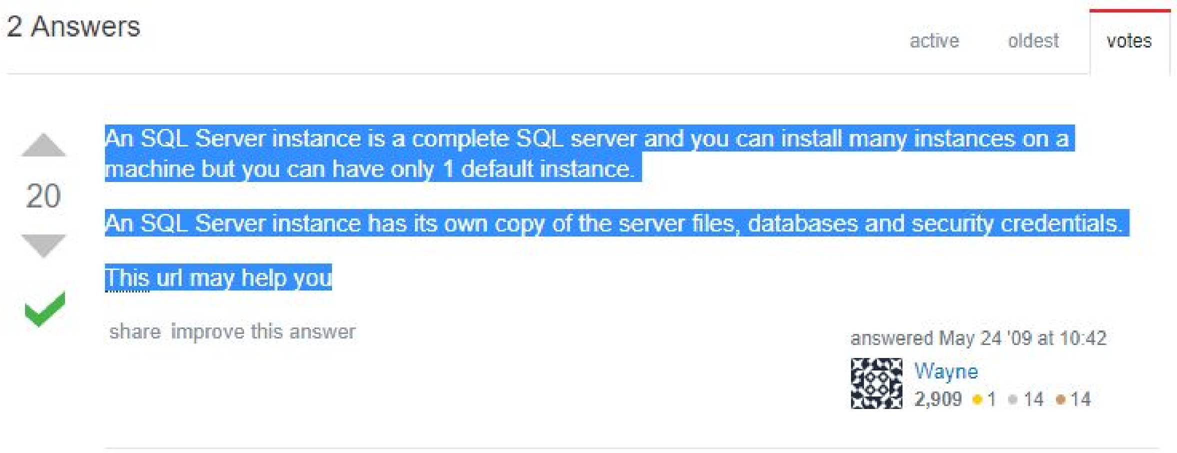
SQL SERVER RESEARCH:

SQL-SERVER INSTANCES:



An SQL Server instance is a complete SQL server and you can install many instances on a machine but you can have only 1 default instance.

An SQL Server instance has its own copy of the server files, databases and security credentials.

This url may help you

SQL-SERVER COLLATIONS:

Collation refers to a set of rules that determine how data is sorted and compared. Character data is sorted using rules that define the correct character sequence, with options for specifying case-sensitivity, accent marks, kana character types and character width.

Case sensitivity:
If A and a, B and b, etc. are treated in the same way then it is case-insensitive. A computer treats A and a differently because it uses ASCII code to differentiate the input. The ASCII value of A is 65, while a is 97. The ASCII value of B is 66 and b is 98.

If **a** and **á**, **o** and **ó** are treated in the same way, then it is accent-insensitive. A computer treats **a** and **á** differently because it uses ASCII code for differentiating the input. The ASCII value of **a** is **97** and **á**is **225**. The ASCII value of **o** is **111** and ó is **243**.

Kana Sensitivity
When Japanese kana characters Hiragana and Katakana are treated differently, it is called Kana sensitive.

Width sensitivity
When a single-byte character (half-width) and the same character when represented as a double-byte character (full-width) are treated differently then it is width sensitive.

Database, Tables and columns with different collation

SQL Server 2000 allows the users to create databases, tables and columns in different collations.

Databases with different collation

```
use master
go
create database BIN collate Latin1_General_BIN
go
create database CI_AI_KS collate Latin1_General_CI_AI_KS
go
create database CS_AS_KS_WS collate Latin1_General_CS_AS_KS_WS
go
```

Tables and columns with different collation

```
Create table Mytable (
[colu] char(10) COLLATE Albanian_CI_AI_KS_WS NULL,
[Maydate] [char] (8) COLLATE Korean_Wansung_Unicode_CS_AS_KS NOT NULL ,
[Risk_Rating] [char] (2) COLLATE SQL_Latin1_General_CP1_CI_AS NOT NULL
)
```

Comparing characters on the databases with different collation

When we run the below code in CI_AI_KS and CS_AS_KS_WS the results will be completely different.

```
declare @Accentvar1 char(1)
declare @Accentvar2 char(1)
declare @Casevar1 char(1)
declare @Casevar2 char(1)
set @casevar1 ='A'
set @casevar2 ='a'
set @Accentvar1 ='a'
set @Accentvar2 ='á'

if @casevar1 = @casevar2
        begin
                print "A and a are treated same"
        end
        else
        begin
                print "A and a are not treated same"
        end

if @Accentvar1 = @Accentvar2
        begin
                print "A and á are treated same"
        end
        else
        begin
                print "A and á are not  treated same"
        end
```

When we execute these statements on a CI_AI_KS database, the results are similar to those shown below.

```
        A and a are treated same
        A and á are treated same
```

When we execute these statements on a CS_AS_KS_WS database, the results are similar to those shown below.

```
        A and a are not treated same
        A and á are not treated same
```

Simulating case sensitivity in a case insensitive database

It is often necessary to simulate case sensitivity in a case insensitive database. The example below shows how you can achieve that.

```
Use CI_AI_KS
go
declare @var1 varchar(10)
declare @var2 varchar(10)
set @var1 ='A'
set @var2 ='a'
if ASCII(@var1) = ASCII(@var2)
print "A and a are treated same"
else
print "A and a are not same"
```

However, the function ASCII cannot be used for words. In order to achieve the same functionality of simulating case sensitiveness, we can use the varbinary data type.

```
Use CI_AI_KS
go
declare @var1 varchar(10)
declare @var2 varchar(10)
set @var1 ='Good'
set @var2 ='gooD'
if cast(@var1 as varbinary) = cast(@var2 as varbinary)
print "Good and gooD are treated same"
else
print "Good and gooD are not treated same"
```

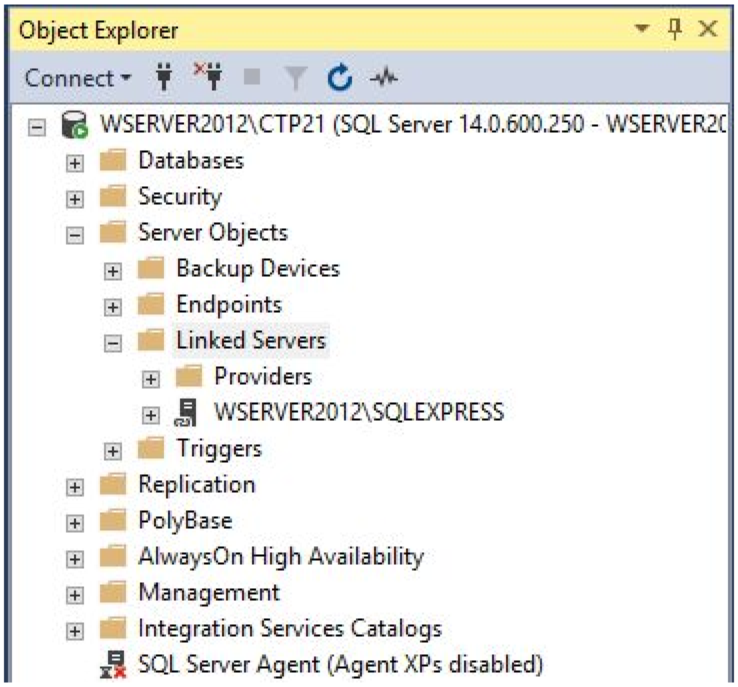
SQL-SERER LINKED SERVER

How to create and configure a linked server in SQL Server Management Studio:

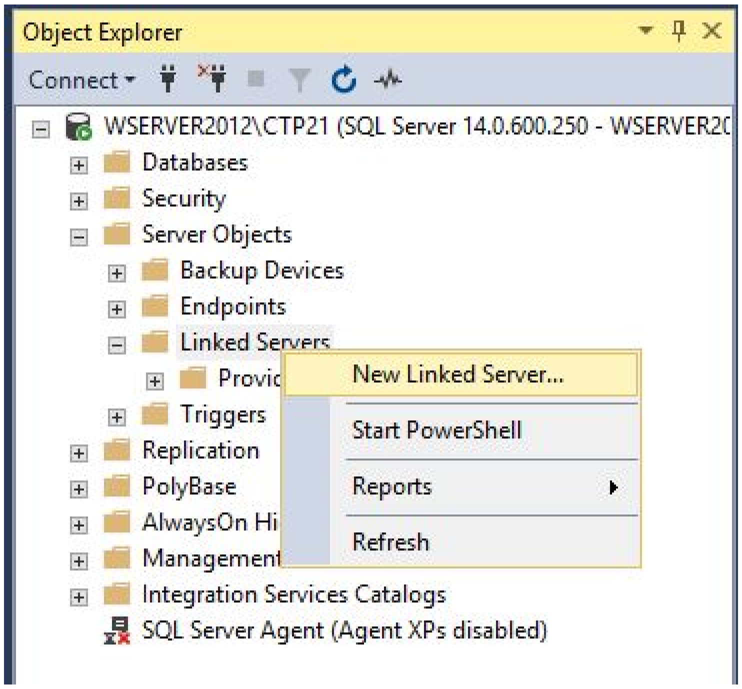
Linked servers allow submitting a T-SQL statement on a SQL Server instance, which returns data from other SQL Server instances. A linked server allows joining data from several SQL Server instances using a single T-SQL statement when data exists on multiple databases on different SQL instances. By using a linked server to retrieve data from several SQL instances, the only thing that should be done is to connect to one SQL instance.

There are two ways of configuring linked server in SSMS. One way is by using sp_addlinkedserver system stored procedure and another is by using SQL Server Management Studio (SSMS) GUI interface.

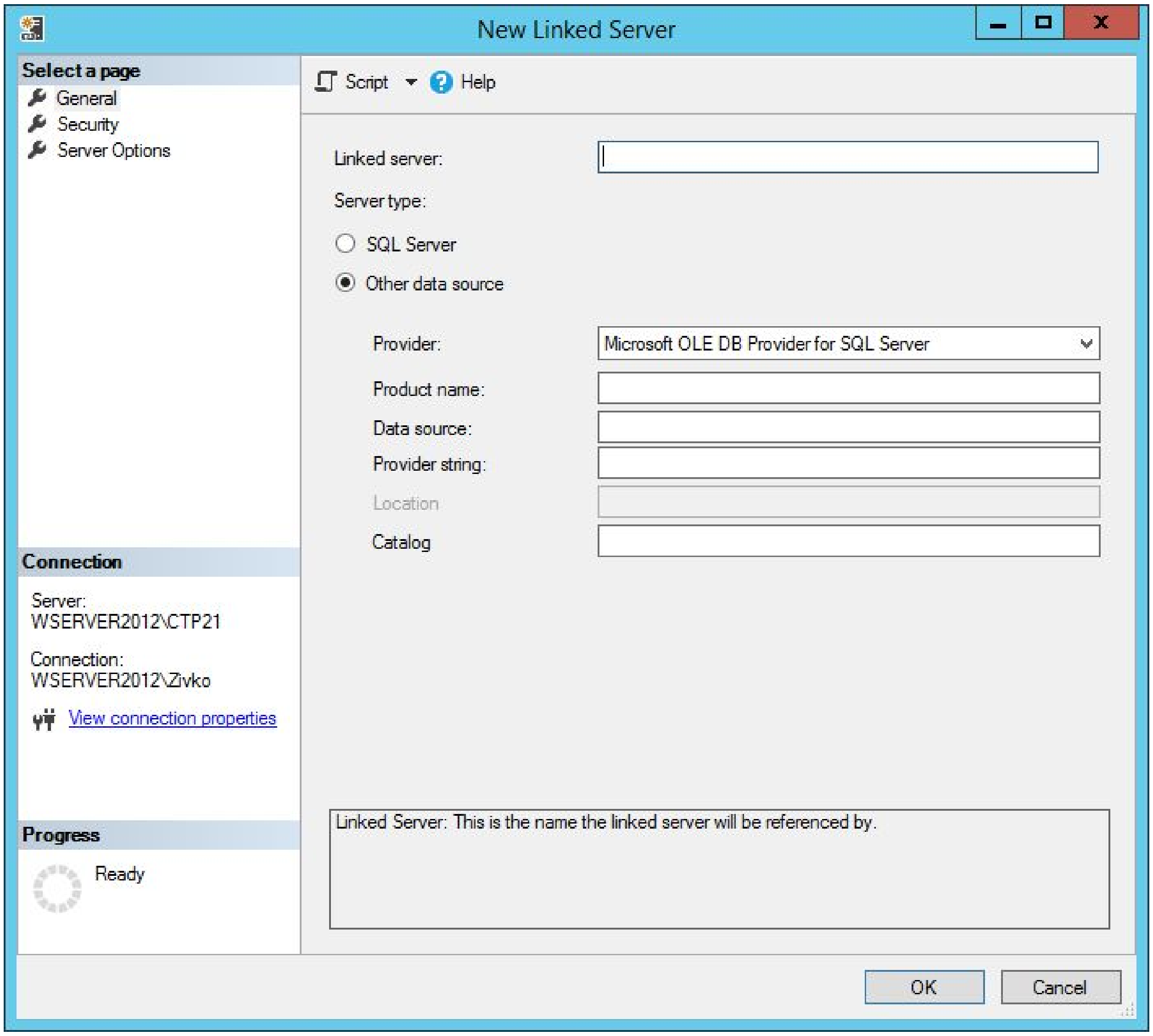
To see all created linked servers in SSMS, under **Object Explorer**, chose the **Server Objects** folder and expand the **Linked Servers** folder:



To create a linked server in SSMS, right click on the **Linked Servers** folder and from the context menu select the **New Linked Server** option:



The **New Linked Server** dialog appears:


New Linked Server

Select a page
- General
- Security
- Server Options

Script ▾ Help

Linked server:

Server type:

( ) SQL Server

(●) Other data source

Provider: Microsoft OLE DB Provider for SQL Server

Product name:

Data source:

Provider string:

Location

Catalog

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

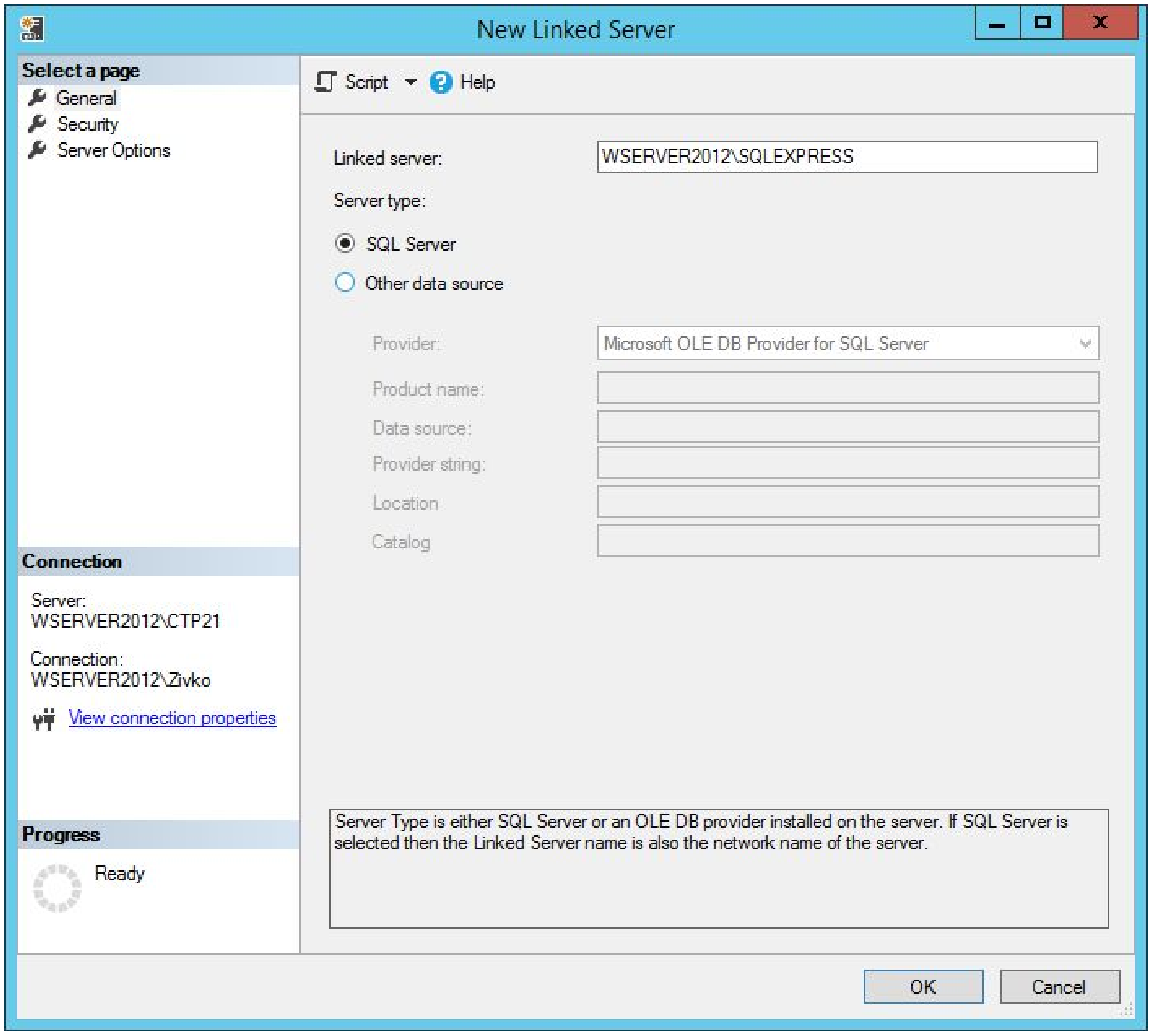
Progress

Ready

Linked Server: This is the name the linked server will be referenced by.

OK Cancel

In this dialog, the name of a linked server and server type must be identified. The linked servers can be defined for different kind of data source if the **Other data source** radio button is chosen. For the purpose of this article, the SQL Server radio button under the **Server type** section will be chosen and in the **Linked server** text box, a name of the server will be entered:



Linked server: WSERVER2012\SQLEXPRESS

Server type:

[✓ SQL Server]

Other data source

Provider: Microsoft OLE DB Provider for SQL Server

Product name:

Data source:

Provider string:

Location

Catalog

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

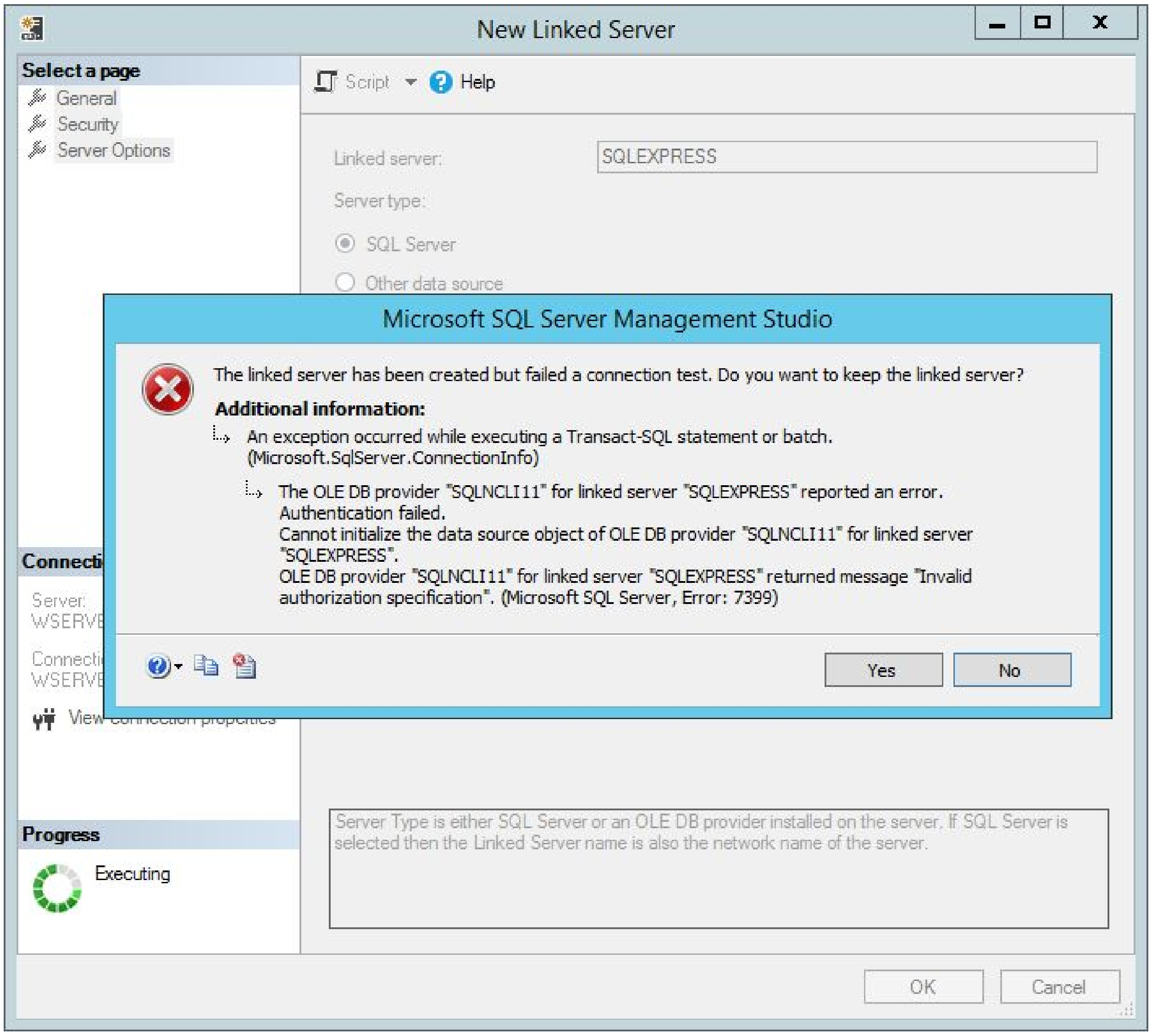
Progress

Ready

Server Type is either SQL Server or an OLE DB provider installed on the server. If SQL Server is selected then the Linked Server name is also the network name of the server.

If the SQL Server type is chosen to configure a SQL Server linked server, the name specified in the **Linked server**text box must be the name of the remote SQL Server.

If a SQL Server instance is default instance, type the name of the computer that hosts the instance of SQL Server**(e.g. WSERVER2012)**. If the SQL Server is a named instance, type the name of the computer and the name of the instance separated by slash (**e.g. WSERVER2012\SQLEXPRESS**).



Microsoft SQL Server Management Studio

The linked server has been created but failed a connection test. Do you want to keep the linked server?

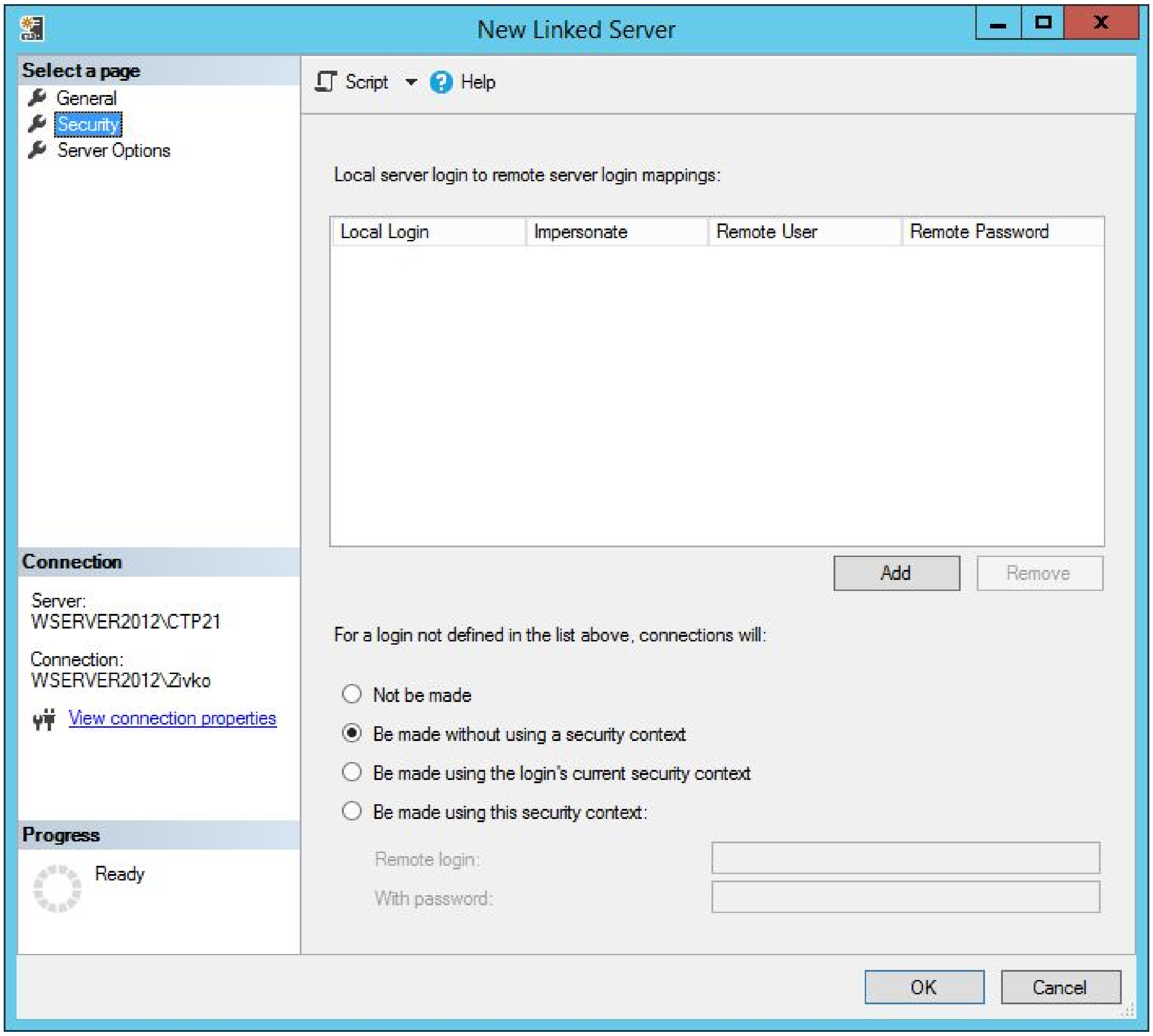
Additional information:

An exception occurred while executing a Transact-SQL statement or batch. (Microsoft.SqlServer.ConnectionInfo)

The OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS" reported an error. Authentication failed.
Cannot initialize the data source object of OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS".
OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS" returned message "Invalid authorization specification". (Microsoft SQL Server, Error: 7399)

Server Type is either SQL Server or an OLE DB provider installed on the server. If SQL Server is selected then the Linked Server name is also the network name of the server.

To set how a user would authenticate to the **WSERVER2012\SQLEXPRESS** instance, under the **Select a page**section on upper left corner of the **New Linked Server** dialog, select the **Security** item:


New Linked Server

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

Ready

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password

For a login not defined in the list above, connections will:

- Not be made
- [✓ Be made without using a security context]
- Be made using the login's current security context
- Be made using this security context:

Remote login:

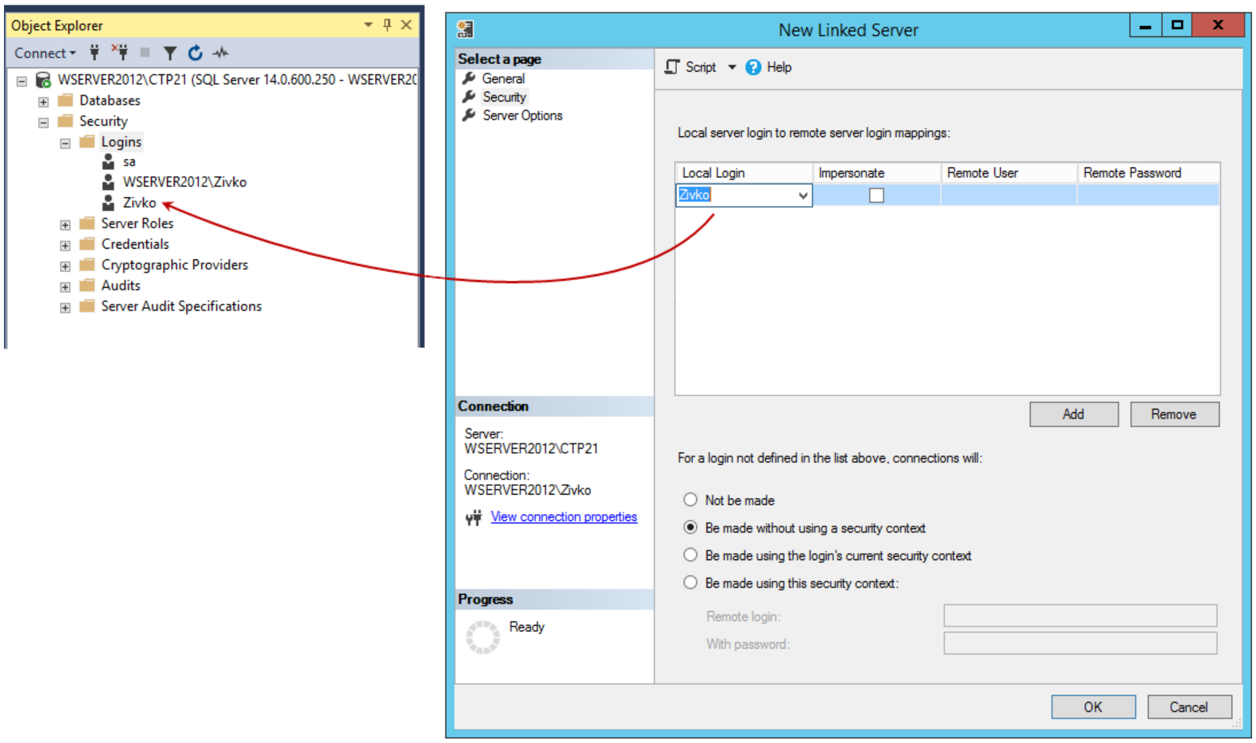
With password:

Here, different ways to authenticate the linked server can be set.

Under the **Local server login to remote server login mappings,** two ways of local logging to a remote login can be set. One way is to associate a local login with a remote login and other way is to impersonate.

Local Login :

In the **Local Login** field, will be listed all the local logins. The local login can be an SQL Server Authentication local login:


Object Explorer

Connect

WSERVER2012\CTP21 (SQL Server 14.0.600.250 - WSERVER20
- Databases
- Security
  - Logins
    - sa
    - WSERVER2012\Zivko
    - Zivko
  - Server Roles
  - Credentials
  - Cryptographic Providers
  - Audits
  - Server Audit Specifications

New Linked Server

Select a page: General, Security, Server Options

Script · Help

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
Zivko	☐		

Add · Remove

For a login not defined in the list above, connections will:

- ( ) Not be made
- (●) Be made without using a security context
- ( ) Be made using the login's current security context
- ( ) Be made using this security context:

Remote login:

With password:

Connection

Server:
WSERVER2012\CTP21

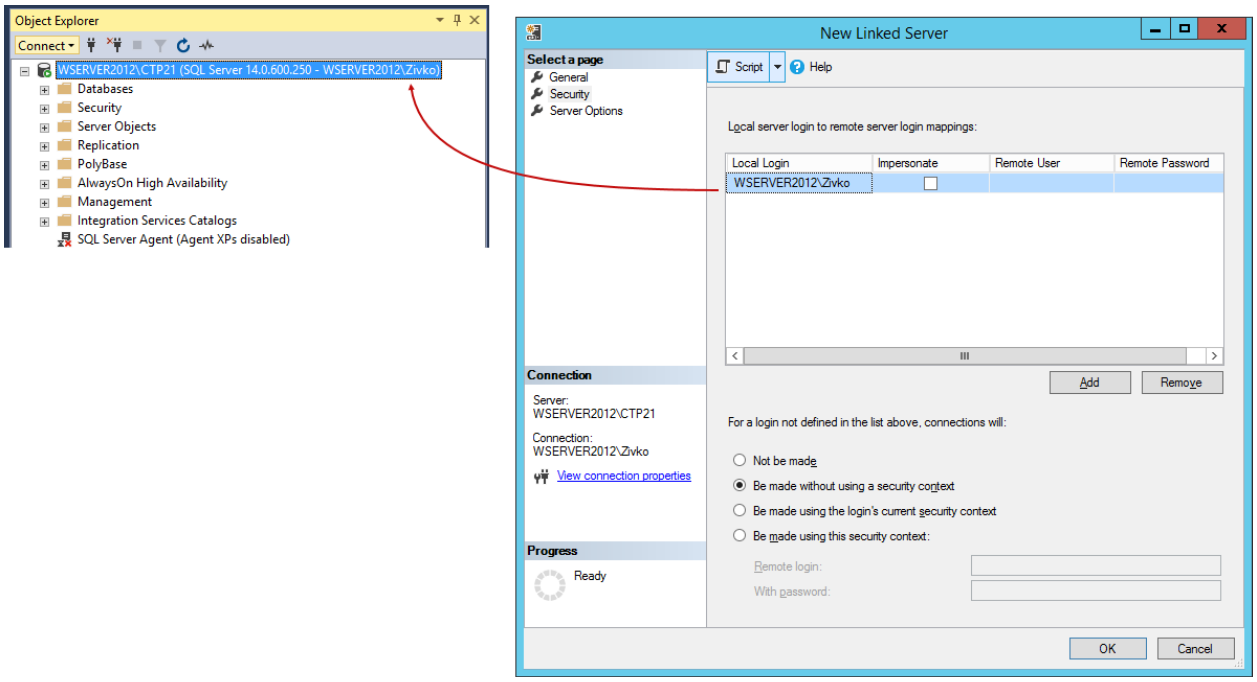
Connection:
WSERVER2012\Zivko

View connection properties

Progress

Ready

OK · Cancel


Object Explorer

Connect

WSERVER2012\CTP21 (SQL Server 14.0.600.250 - WSERVER2012\Zivko)
- Databases
- Security
- Server Objects
- Replication
- PolyBase
- AlwaysOn High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent (Agent XPs disabled)

New Linked Server

Select a page: General, Security, Server Options

Script · Help

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	☐		

Add · Remove

For a login not defined in the list above, connections will:

- ( ) Not be made
- (●) Be made without using a security context
- ( ) Be made using the login's current security context
- ( ) Be made using this security context:

Remote login:

With password:

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Progress

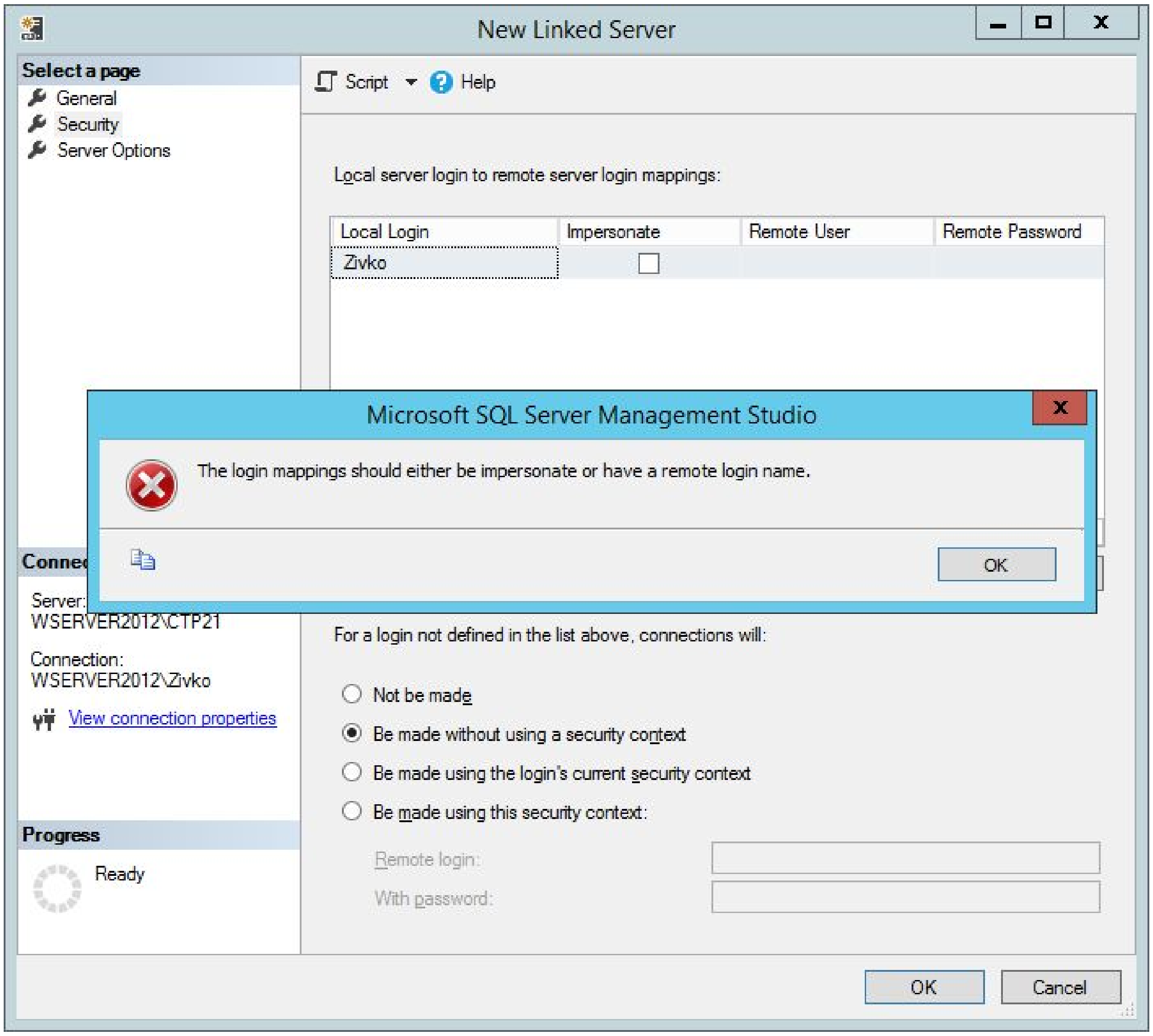
Ready

OK · Cancel

Now, when clicking the **OK** button on the New Linked Server dialog, the following error message will appear:

```
The login mappings should either be impersonate or have a remote login name.
```

See the image below:


New Linked Server
Select a page
General
Security
Server Options
Script
Help
Local server login to remote server login mappings:
Local Login
Impersonate
Remote User
Remote Password
Zivko
Microsoft SQL Server Management Studio
OK
Server:
WSERVER2012\CTP21
Connection:
WSERVER2012\Zivko
View connection properties
Progress
Ready
For a login not defined in the list above, connections will:
Not be made
Be made without using a security context
Be made using the login's current security context
Be made using this security context:
Remote login:
With password:
OK
Cancel

The login mappings should either be impersonate or have a remote login name.

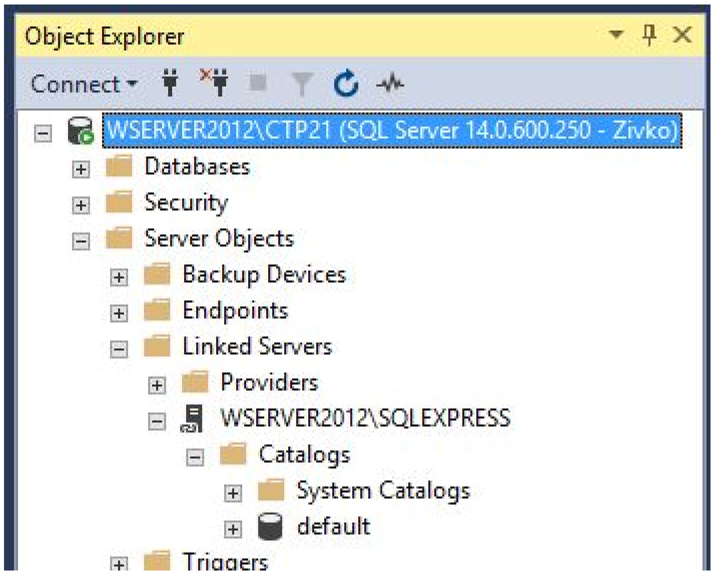
Local Login	Impersonate	Remote User	Remote Password
Zivko	☐		

For a login not defined in the list above, connections will: [✓ Be made without using a security context]

This happens because the **Impersonate** check box isn't checked.

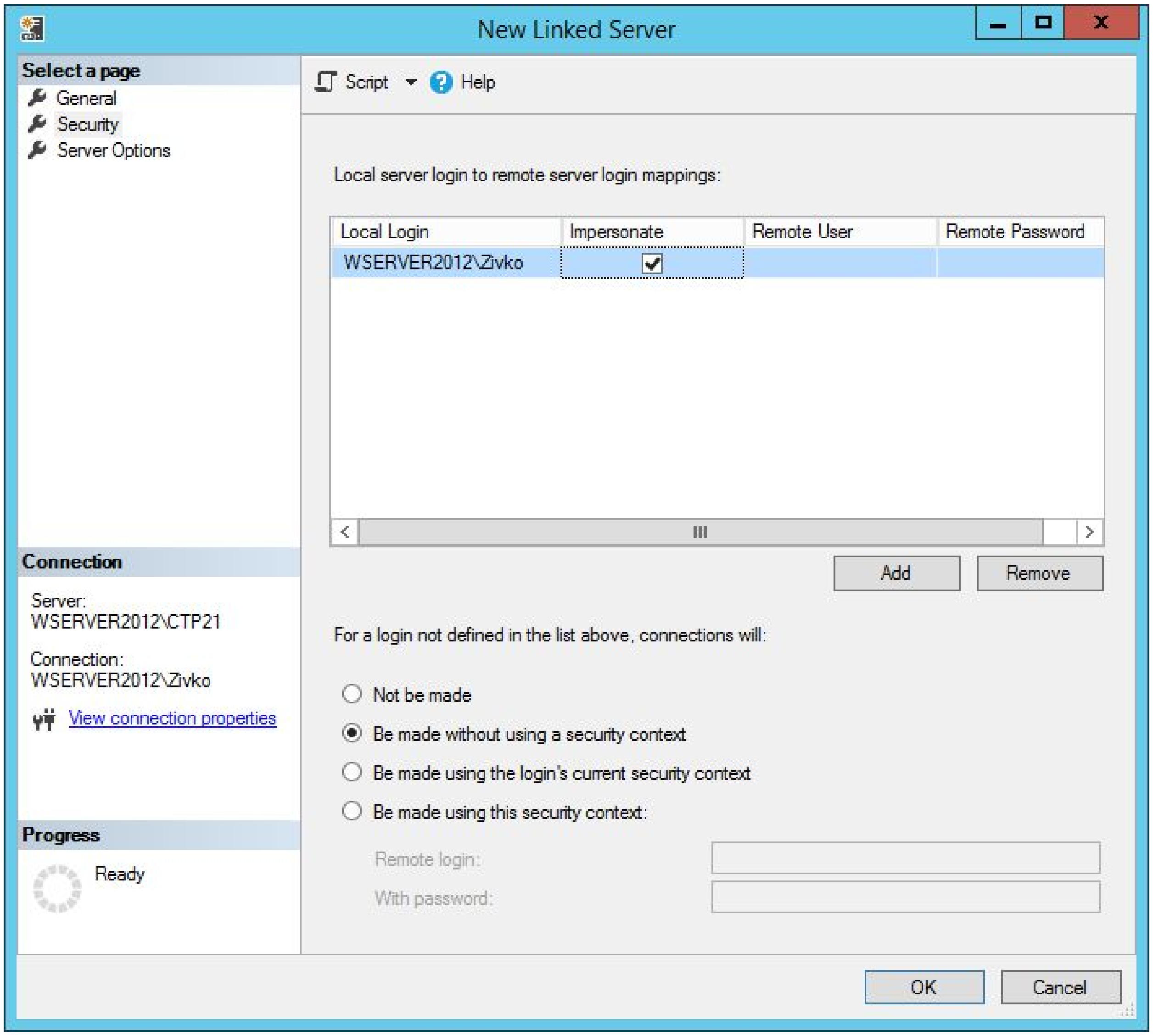
Impersonate

The Impersonate check box when is checked passes the local login credentials to the linked server. For SQL Server Authentication, the same login with the exact credentials must exist on the linked server, otherwise when connected to the server with the SQL Server Authentication, the list of the databases under the **Catalogs** folder may look like this:



or Windows logins, the login must be a valid login on the linked server. In order to use impersonation, the delegation between the local server and the linked server must be set.

Let’s create a linked server using the local Windows login. From the **Local Login** combo box, choose the local Windows login and check the **Impersonate** checkbox and press the OK button:



Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	✓		

For a login not defined in the list above, connections will:

- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

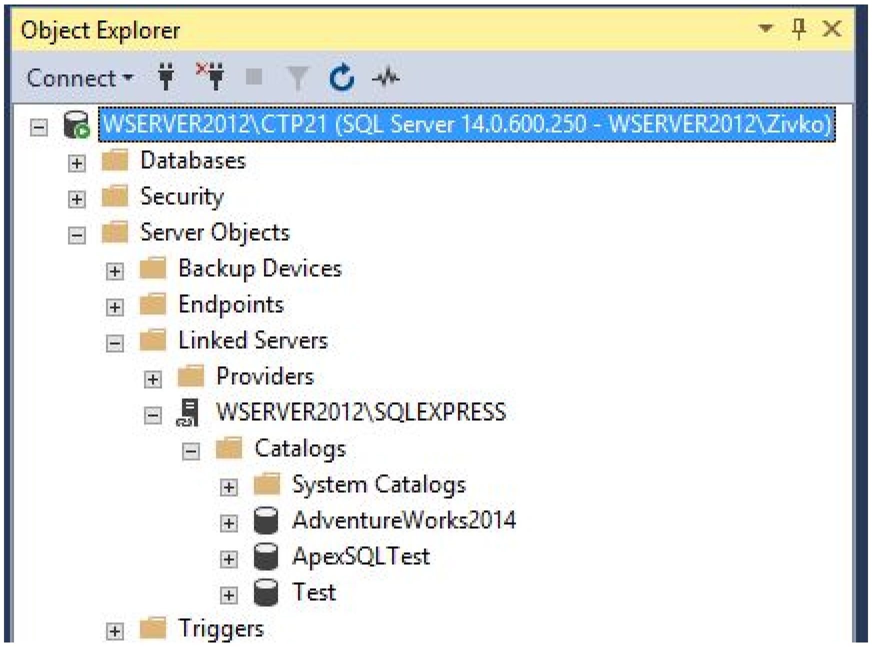
Remote login:
With password:

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

Under the **Catalogs** folder, all databases that are located on the linked server will be listed:



Remote User:

The remote user option allows users from the local SQL server to connect to the linked SQL server even though their credentials aren't present on the remote server by using the credentials from the user that exists on the remote server. Basically, it allows local logins to connect to a remote server as a different login that must exist on a remote server.

How to create and configure a linked server in SQL Server Management Studio

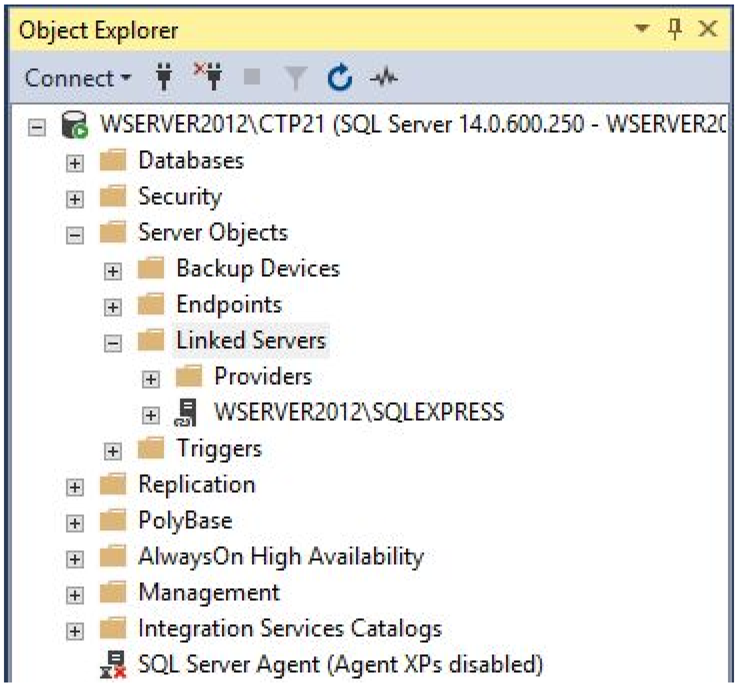
June 9, 2017 by Marko Zivkovic

Linked servers allow submitting a T-SQL statement on a SQL Server instance, which returns data from other SQL Server instances. A linked server allows joining data from several SQL Server instances using a single T-SQL statement when data exists on multiple databases on different SQL instances. By using a linked server to retrieve data from several SQL instances, the only thing that should be done is to connect to one SQL instance.

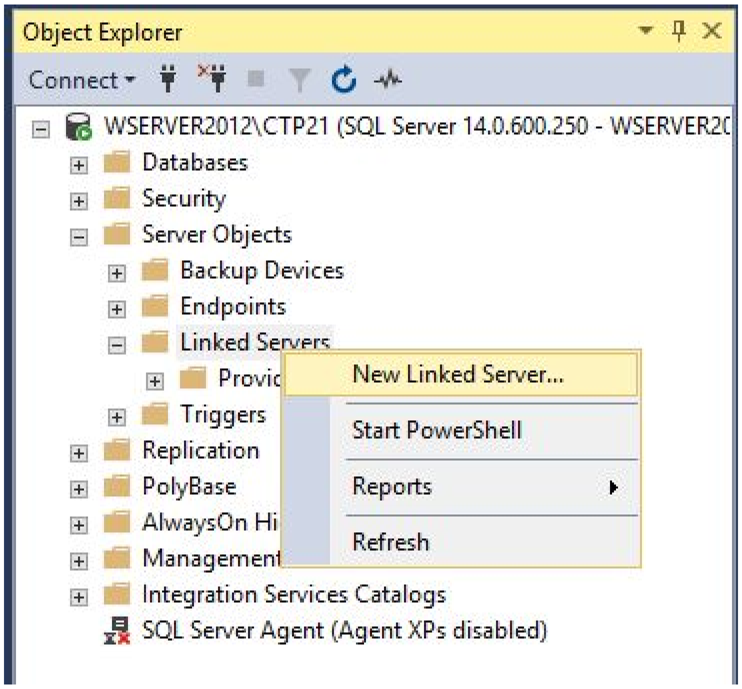
There are two ways of configuring linked server in SSMS. One way is by using sp_addlinkedserver system stored procedure and another is by using SQL Server Management Studio (SSMS) GUI interface.

In this article will be explained how to configure a linked server using a SQL Server data source. More information about other data sources can be found on this link.

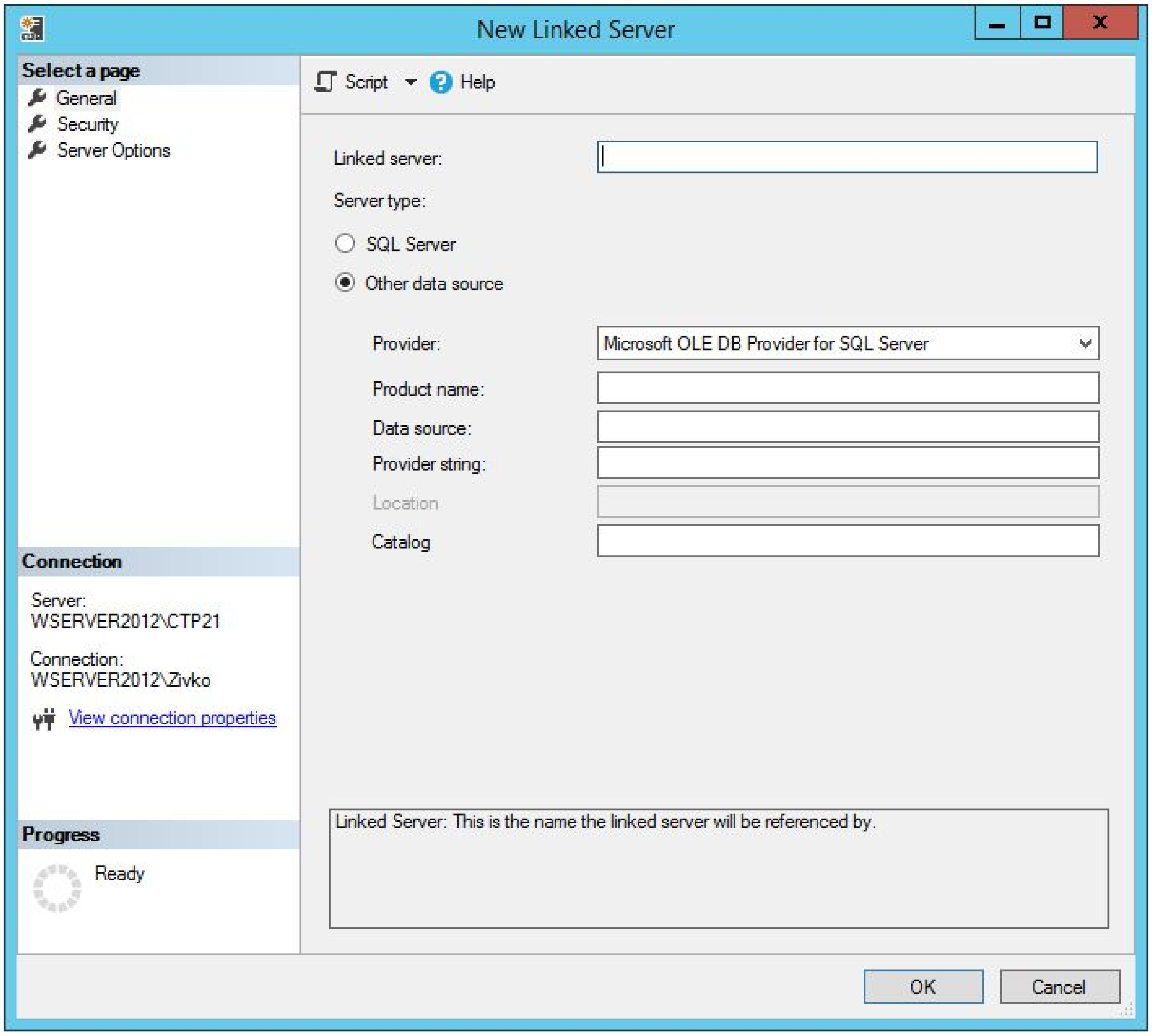
To see all created linked servers in SSMS, under **Object Explorer**, chose the **Server Objects** folder and expand the **Linked Servers** folder:



To create a linked server in SSMS, right click on the **Linked Servers** folder and from the context menu select the **New Linked Server** option:



The **New Linked Server** dialog appears:



Linked server:

Server type:

SQL Server [✓ Other data source]

Provider: Microsoft OLE DB Provider for SQL Server

Product name:

Data source:

Provider string:

Location

Catalog

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

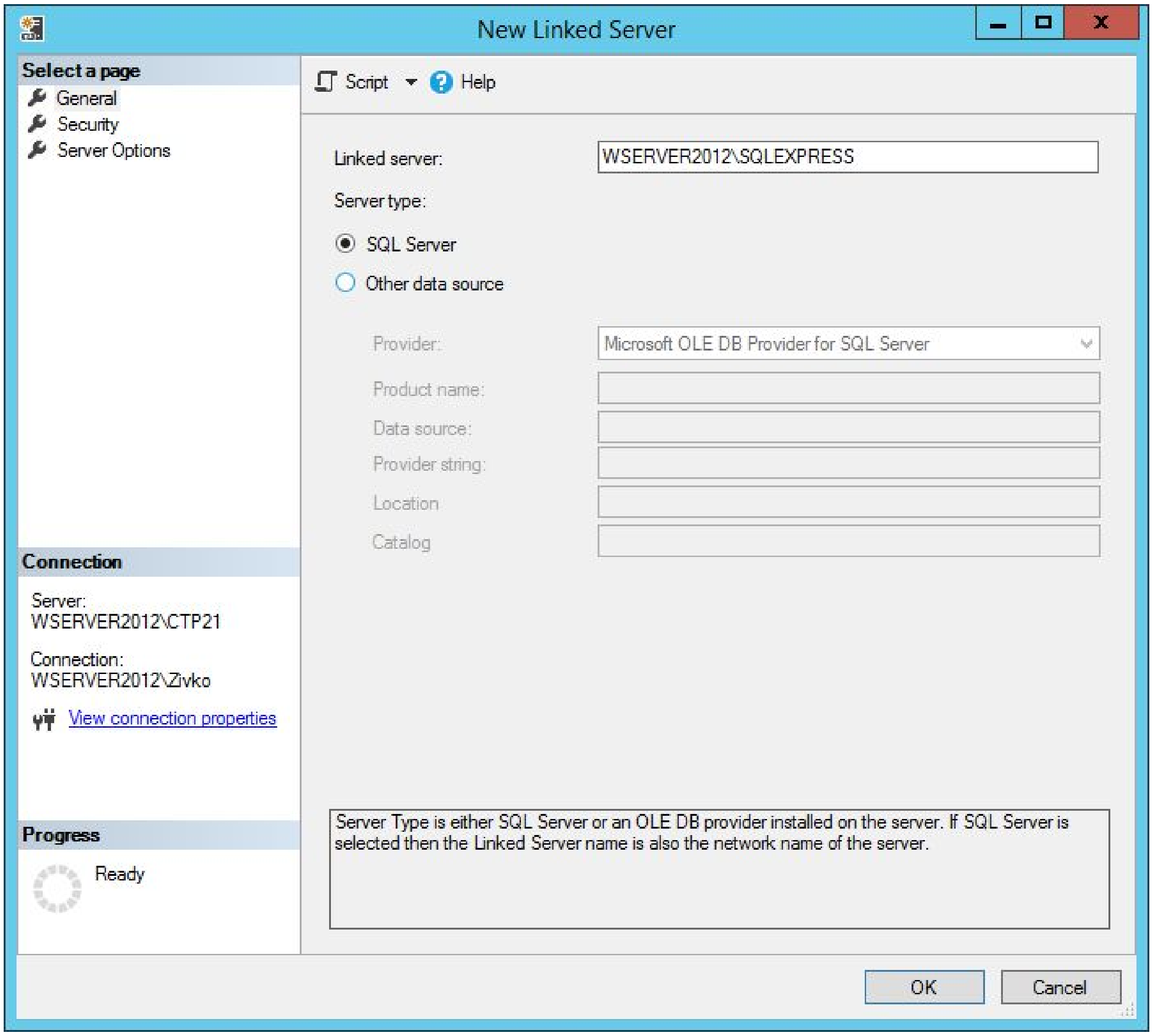
View connection properties

Progress

Ready

Linked Server: This is the name the linked server will be referenced by.

In this dialog, the name of a linked server and server type must be identified. The linked servers can be defined for different kind of data source if the **Other data source** radio button is chosen. For the purpose of this article, the SQL Server radio button under the **Server type** section will be chosen and in the **Linked server** text box, a name of the server will be entered:



Linked server: WSERVER2012\SQLEXPRESS

Server type:

(●) SQL Server

( ) Other data source

Provider: Microsoft OLE DB Provider for SQL Server

Product name:

Data source:

Provider string:

Location

Catalog

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Progress

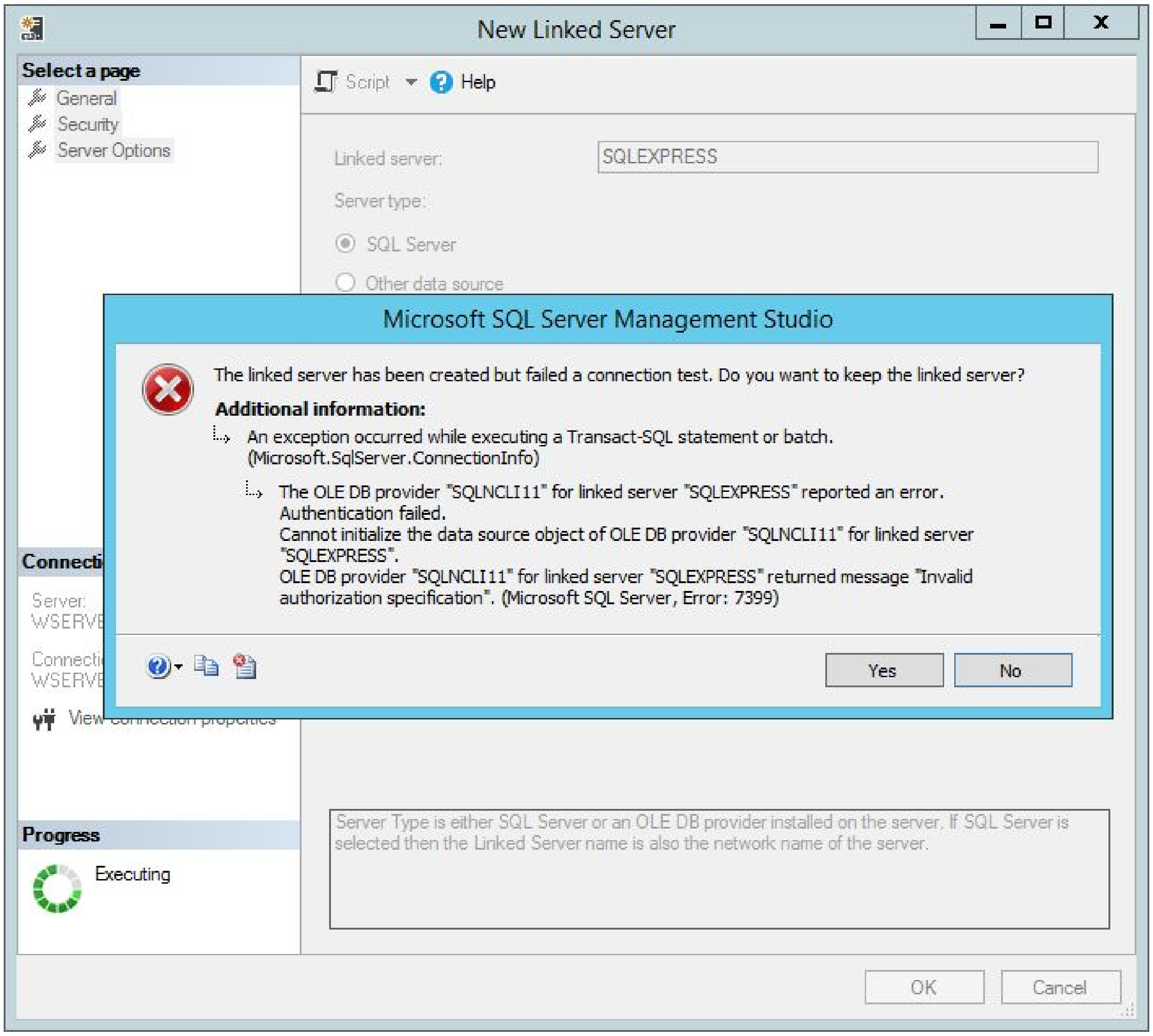
Ready

Server Type is either SQL Server or an OLE DB provider installed on the server. If SQL Server is selected then the Linked Server name is also the network name of the server.

OK Cancel

If the SQL Server type is chosen to configure a SQL Server linked server, the name specified in the **Linked server**text box must be the name of the remote SQL Server.

If a SQL Server instance is default instance, type the name of the computer that hosts the instance of SQL Server**(e.g. WSERVER2012)**. If the SQL Server is a named instance, type the name of the computer and the name of the instance separated by slash (**e.g. WSERVER2012\SQLEXPRESS**). Otherwise the following error may occur when the **OK** button is pressed:



Microsoft SQL Server Management Studio

The linked server has been created but failed a connection test. Do you want to keep the linked server?

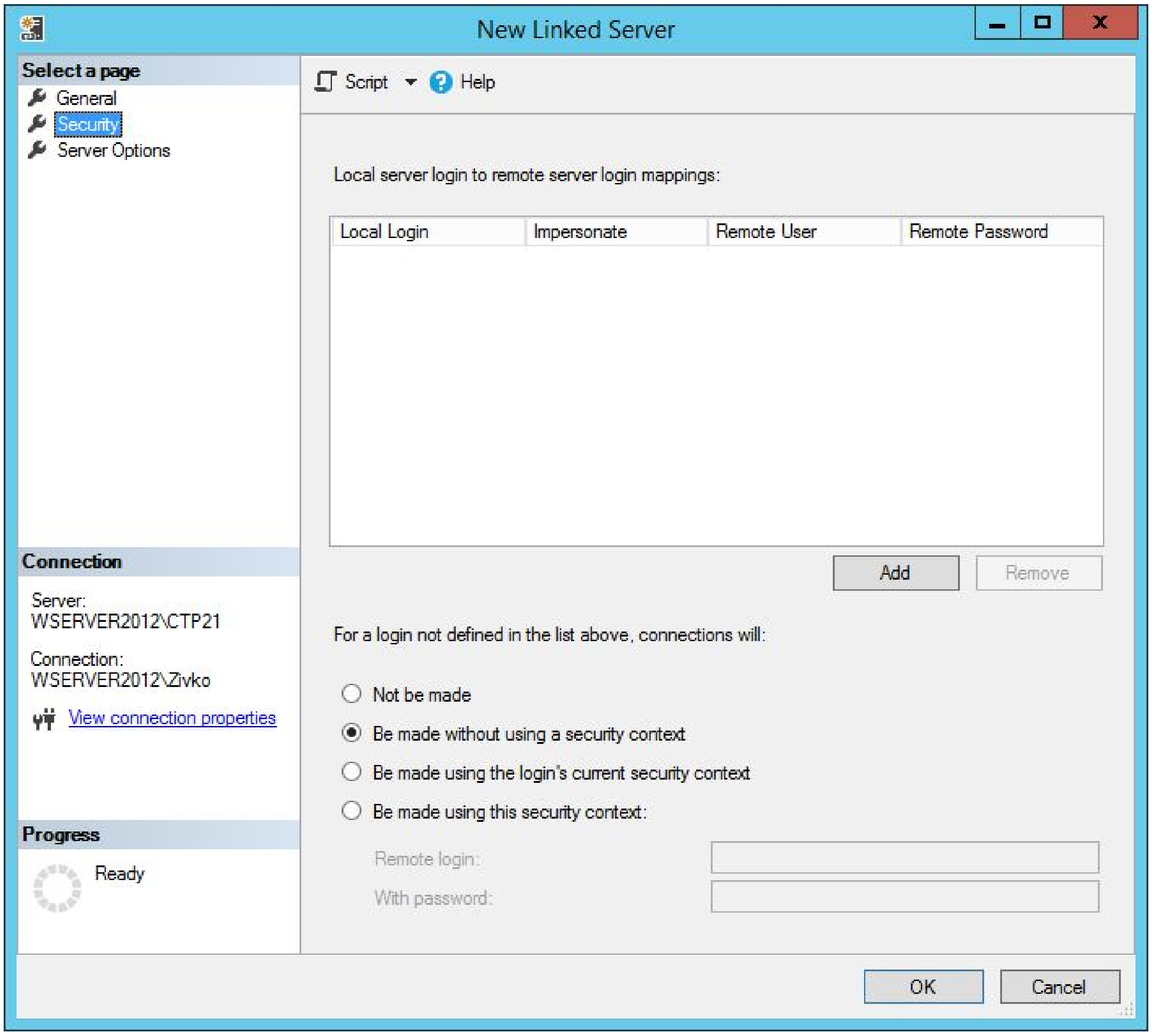
Additional information:

An exception occurred while executing a Transact-SQL statement or batch. (Microsoft.SqlServer.ConnectionInfo)

The OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS" reported an error. Authentication failed.
Cannot initialize the data source object of OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS".
OLE DB provider "SQLNCLI11" for linked server "SQLEXPRESS" returned message "Invalid authorization specification". (Microsoft SQL Server, Error: 7399)

Server Type is either SQL Server or an OLE DB provider installed on the server. If SQL Server is selected then the Linked Server name is also the network name of the server.

To set how a user would authenticate to the **WSERVER2012\SQLEXPRESS** instance, under the **Select a page**section on upper left corner of the **New Linked Server** dialog, select the **Security** item:


New Linked Server

Select a page: General, Security, Server Options

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Progress: Ready

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password

For a login not defined in the list above, connections will:

- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

Remote login:

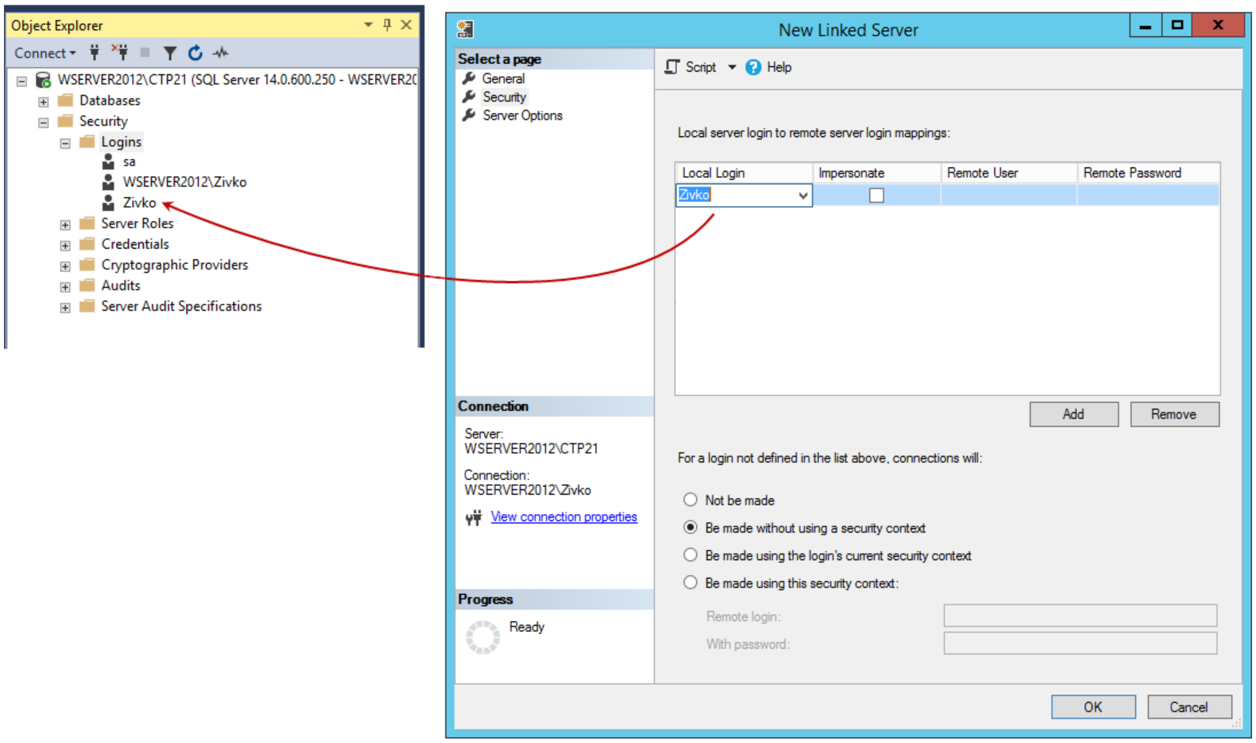
With password:

Here, different ways to authenticate the linked server can be set.

Under the **Local server login to remote server login mappings,** two ways of local logging to a remote login can be set. One way is to associate a local login with a remote login and other way is to impersonate.

Local Login

In the **Local Login** field, will be listed all the local logins. The local login can be an SQL Server Authentication local login:



Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
Zivko	☐		

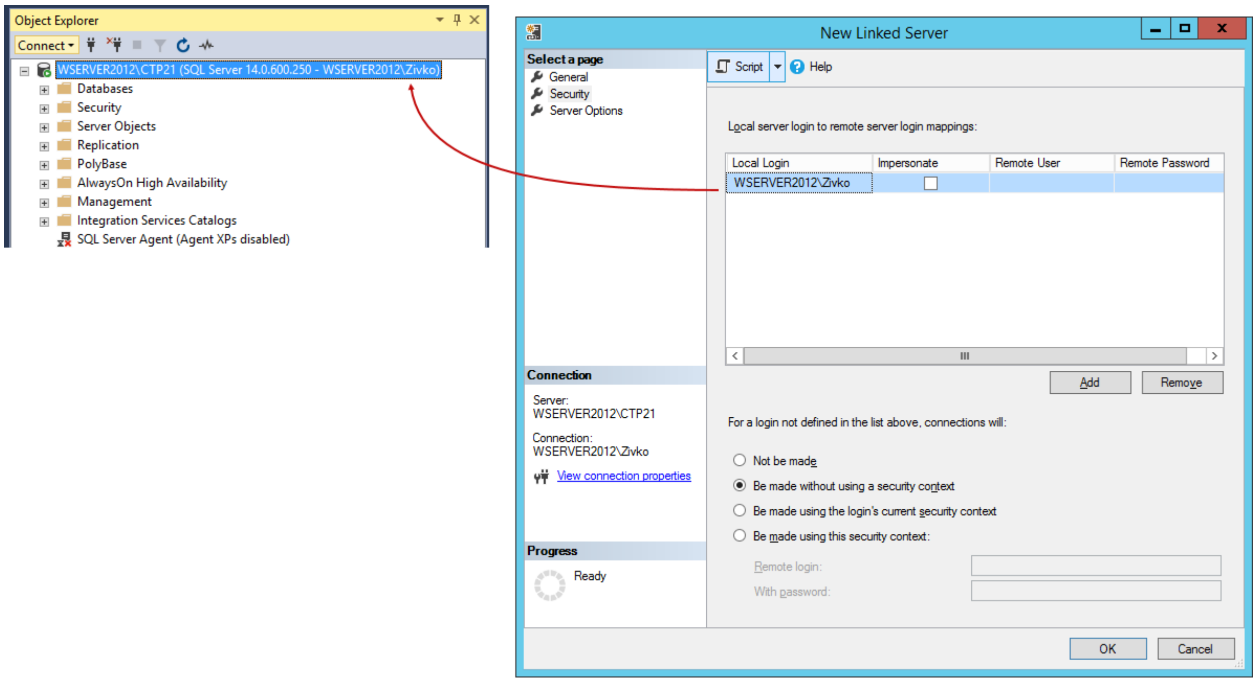
For a login not defined in the list above, connections will:

- Not be made
- [✓ Be made without using a security context]
- Be made using the login's current security context
- Be made using this security context:

Remote login:

With password:

Or a Windows Authentication login:



Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	☐		

For a login not defined in the list above, connections will:

- Not be made
- [✓ Be made without using a security context]
- Be made using the login's current security context
- Be made using this security context:

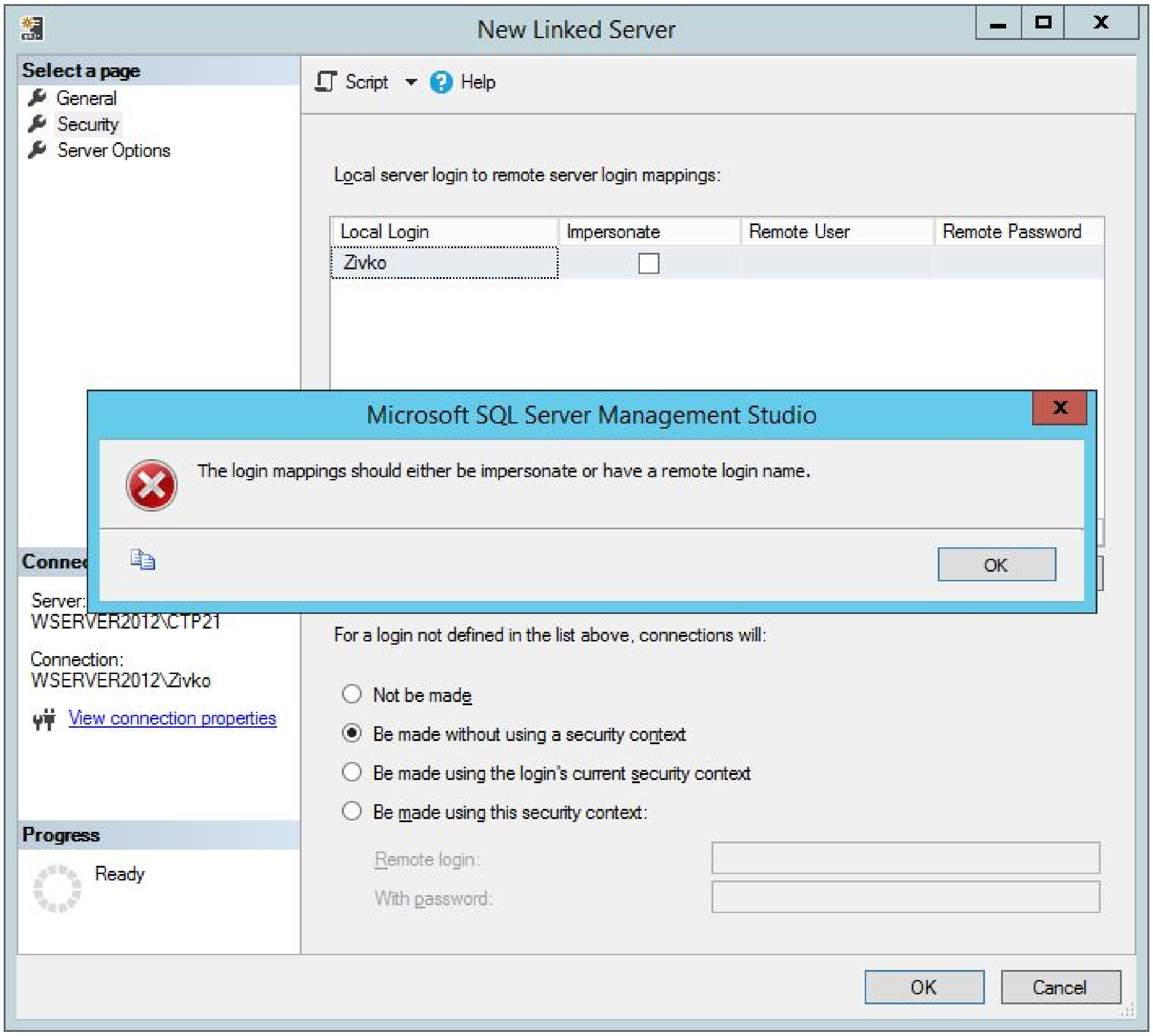
Remote login:

With password:

Now, when clicking the **OK** button on the New Linked Server dialog, the following error message will appear:

```
The login mappings should either be impersonate or have a remote login
name.
```

See the image below:



Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
Zivko	☐		

Microsoft SQL Server Management Studio

The login mappings should either be impersonate or have a remote login name.

For a login not defined in the list above, connections will:

- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

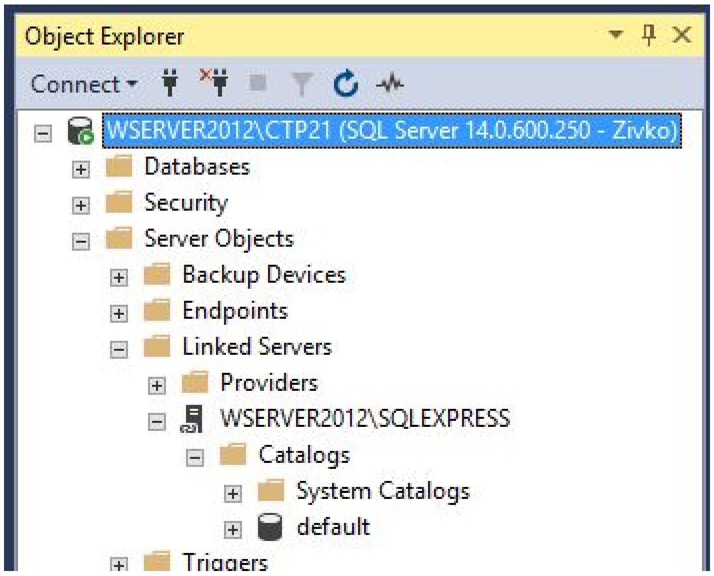
Remote login:

With password:

This happens because the **Impersonate** check box isn't checked.

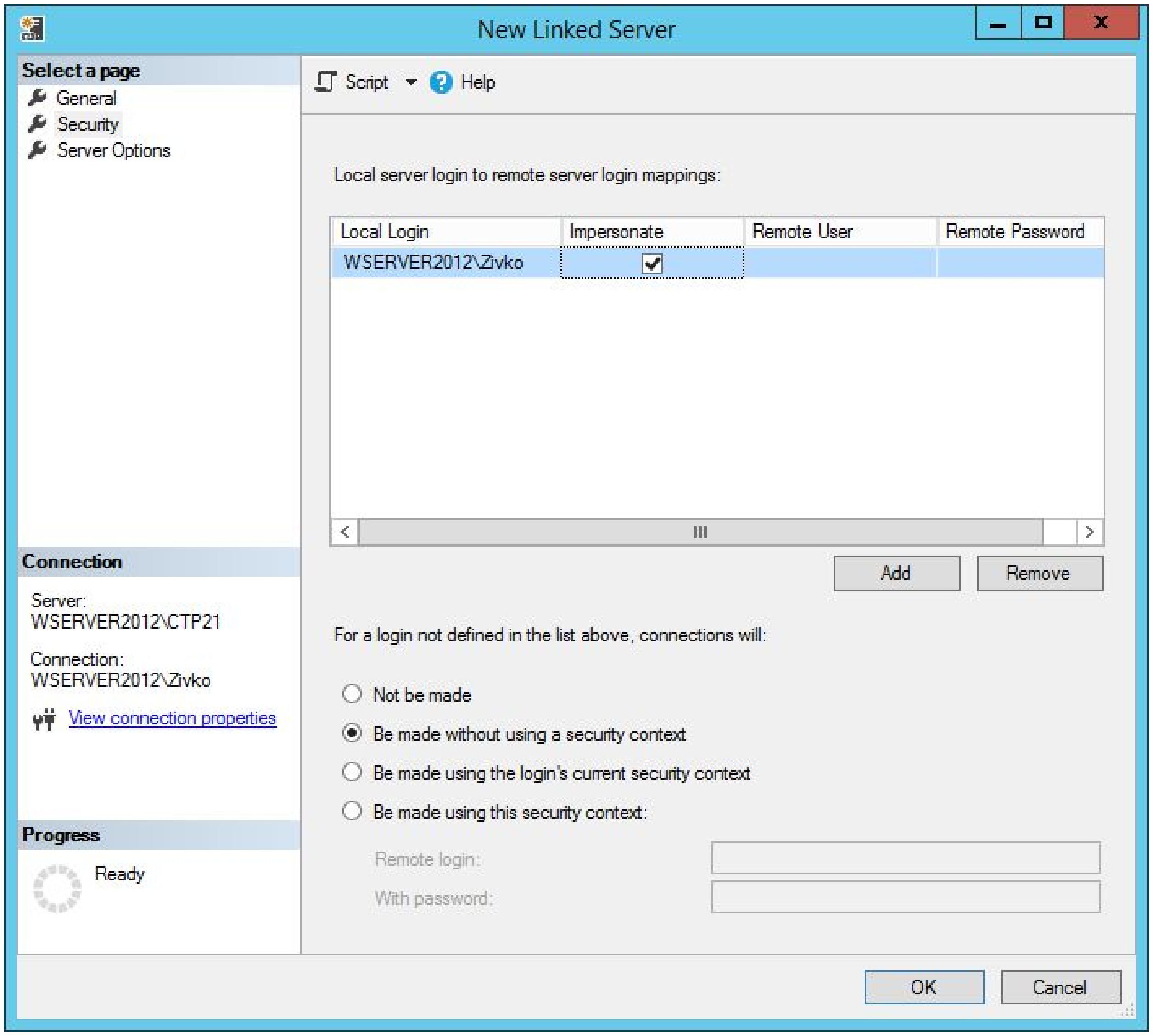
Impersonate

The Impersonate check box when is checked passes the local login credentials to the linked server. For SQL Server Authentication, the same login with the exact credentials must exist on the linked server, otherwise when connected to the server with the SQL Server Authentication, the list of the databases under the **Catalogs** folder may look like this:



For Windows logins, the login must be a valid login on the linked server. In order to use impersonation, the delegation between the local server and the linked server must be set.

Let's create a linked server using the local Windows login. From the **Local Login** combo box, choose the local Windows login and check the **Impersonate** checkbox and press the OK button:



Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	✓		

For a login not defined in the list above, connections will:

- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

Remote login:

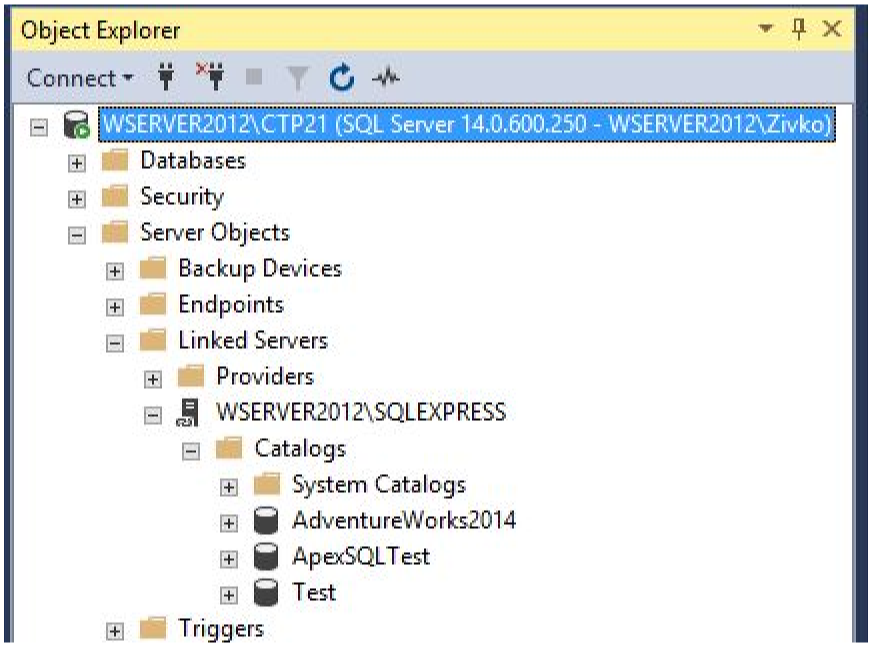
With password:

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

Under the **Catalogs** folder, all databases that are located on the linked server will be listed:



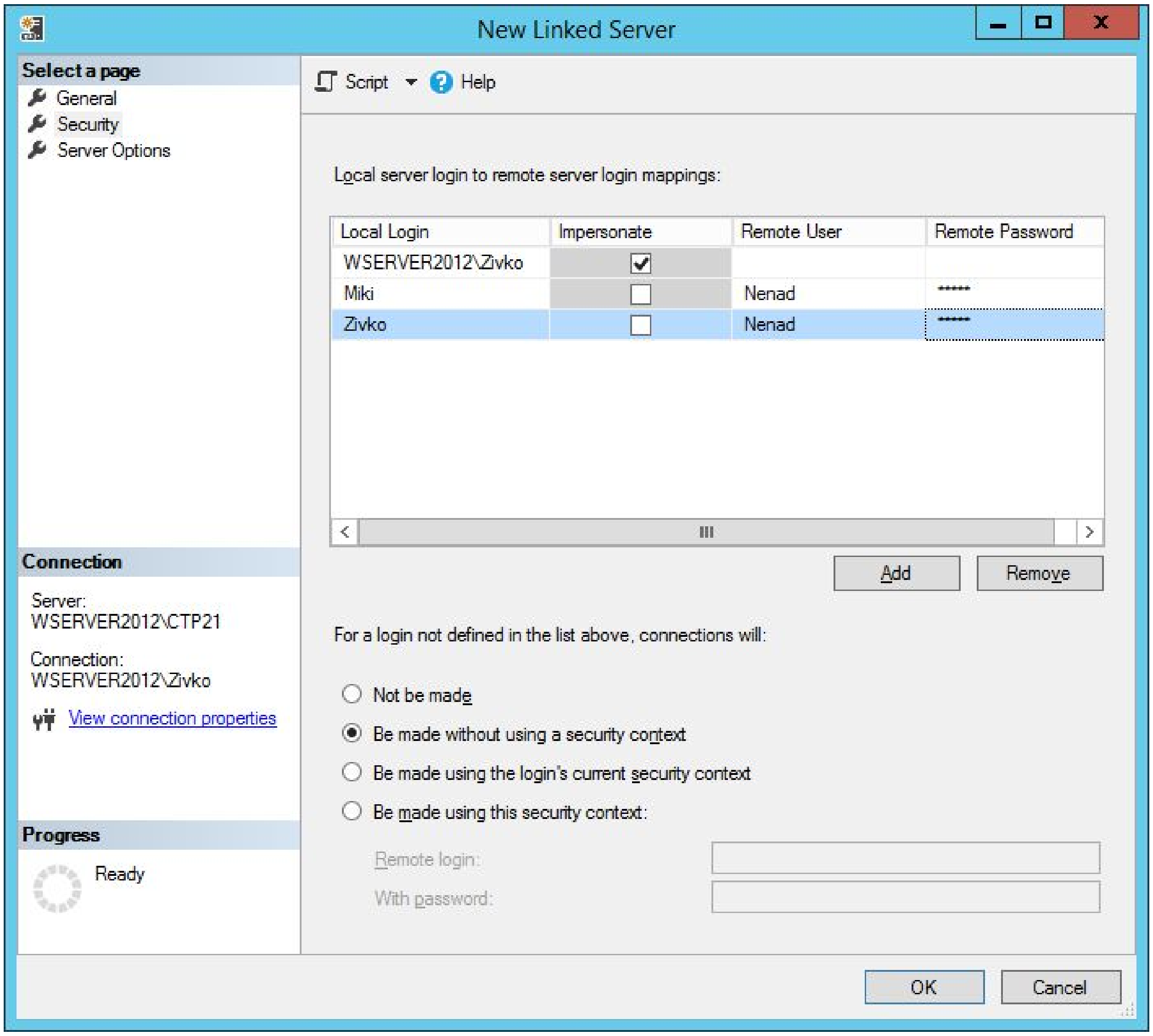
Remote User

The remote user option allows users from the local SQL server to connect to the linked SQL server even though their credentials aren’t present on the remote server by using the credentials from the user that exists on the remote server. Basically, it allows local logins to connect to a remote server as a different login that must exist on a remote server.

Remote Password:

Specify the password of the remote user.

From the **Local Login** drop down list, choose a local login which should map to a remote login. On the **Remote User** field, enter the name of the remote user that exists on the remote server and in the **Remote Password** filed, enter a password of that remote user. Then, press the **OK** button


New Linked Server

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

Progress

Ready

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	✓		
Miki	☐	Nenad	******
Zivko	☐	Nenad	******

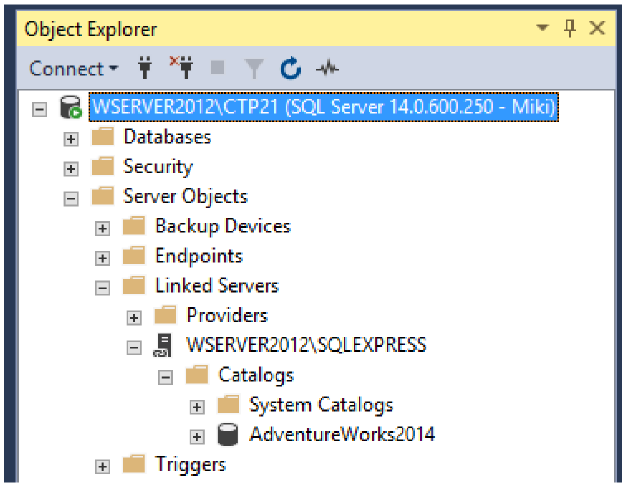
For a login not defined in the list above, connections will:

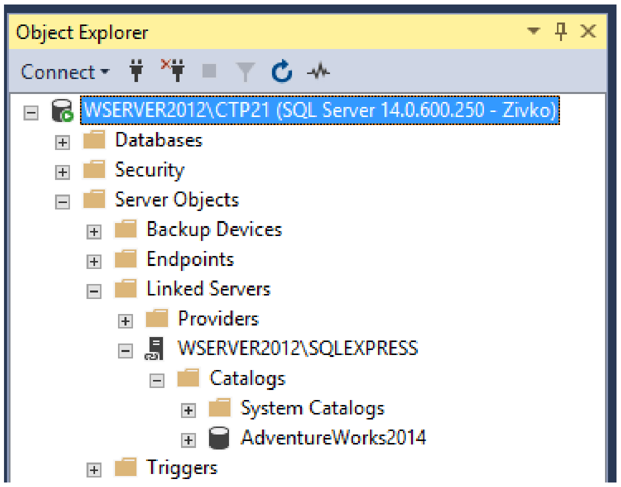
- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

Remote login:

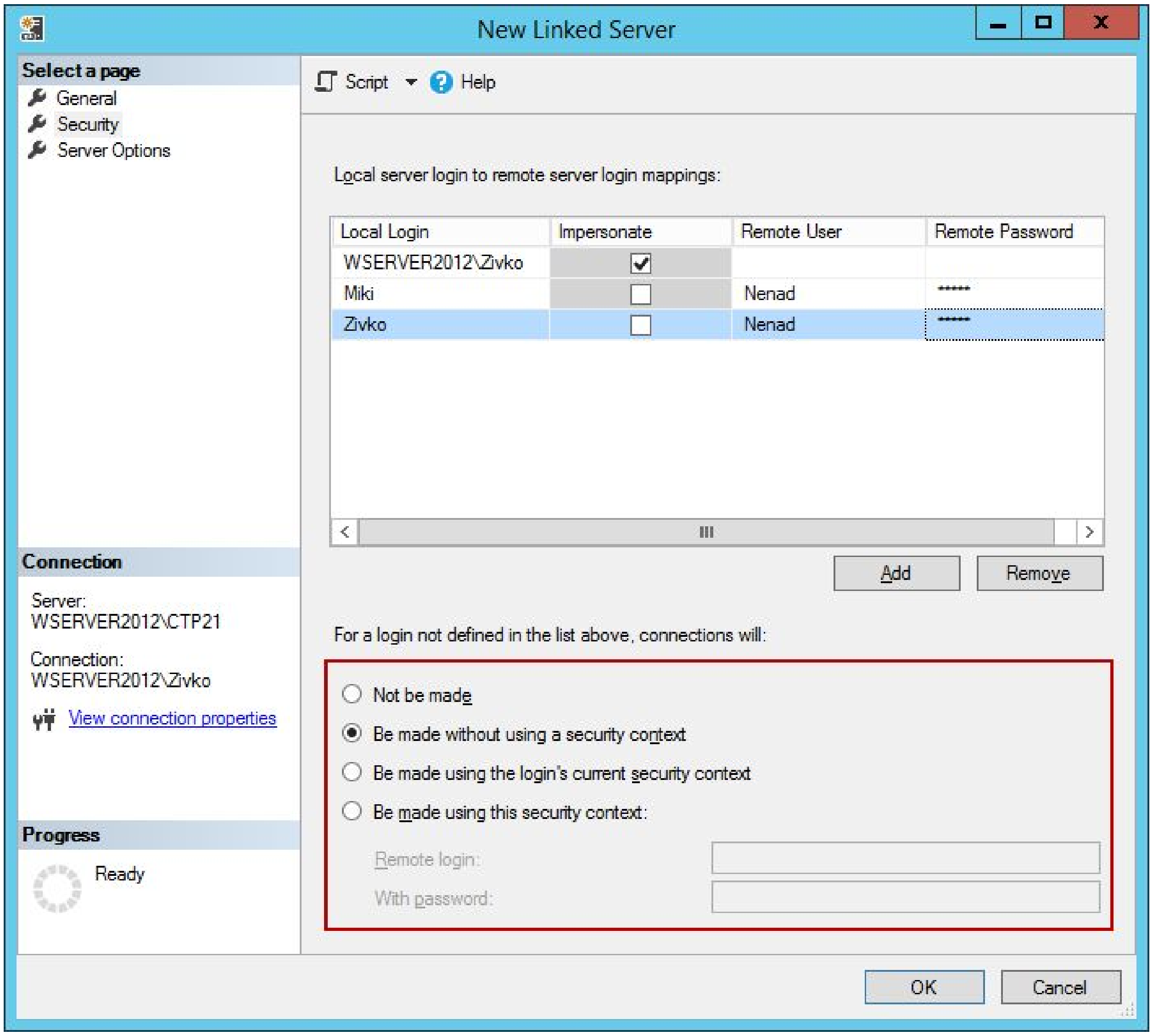
With password:

Now, when connected to the local server using SQL Server Authentication, with Miki or Zivko credentials, under the Catalogs folder, all databases that are available on a remote server for the Nenad remote login will be listed:





Additionally, on the Linked Server dialog, it can be identified how logins that are not set in the **Local server login to remote server login mappings** list will connect to the linked server, for that there are four options that can be used and they are located under the **For a login not defined in the list above, connections will** section:



Server: WSERVER2012\CTP21

Connection: WSERVER2012\Zivko

Ready

Local server login to remote server login mappings:

Local Login	Impersonate	Remote User	Remote Password
WSERVER2012\Zivko	✓		
Miki	☐	Nenad	*****
Zivko	☐	Nenad	*****

For a login not defined in the list above, connections will:

- ○ Not be made
- ● Be made without using a security context
- ○ Be made using the login's current security context
- ○ Be made using this security context:

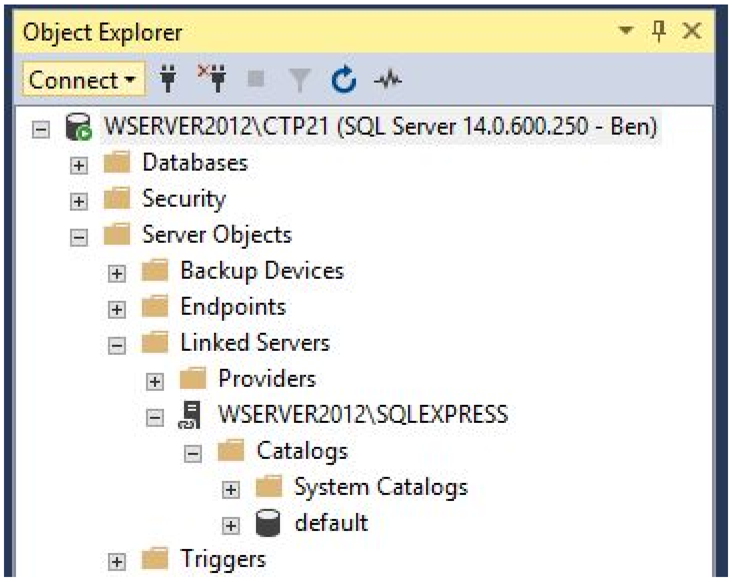
Remote login:

With password:

Not be made

If this radio button is chosen, any users that aren't identified in the **Local server login to remote server login mappings** list cannot establish connection to the linked server.

For example, if login with different account (e.g. Ben) that not set in the login mapping list the list of the databases under the Catalogs folder will look like this:



If you attempt to execute a linked server query:

```
:LECT * FROM
 
 /SERVER2012\SQLEXPRESS].AdventureWorks2014.HumanResources.Employee
```

The following result will appear:

```
Msg 7416, Level 16, State 1, Line 1
Access to the remote server is denied because no login-mapping exists.
```

Be made without using a security context

The **Be made without using a security context** option is used for connecting to data sources that do not require any authentication, for example like a text file. When this option is selected for connect to a linked server, will have the same effect as selecting the “Not be made” option.

If you attempt to execute a linked server query:

```
:LECT * FROM
 
 /SERVER2012\SQLEXPRESS].AdventureWorks2014.HumanResources.Employee
```

The following message e may appear:

```
OLE DB provider “SQLNCLI11” for linked server “WSERVER2012\SQLEXPRESS”
returned message “Invalid authorization specification”.
Msg 7399, Level 16, State 1, Line 1
The OLE DB provider “SQLNCLI11” for linked server “WSERVER2012\SQLEXPRESS”
reported an error. Authentication failed.
Msg 7303, Level 16, State 1, Line 1
Cannot initialize the data source object of OLE DB provider “SQLNCLI11”
for linked server “WSERVER2012\SQLEXPRESS”.
```

Be made using the login’s current security context

If this option is chosen, it will pass the current security context of the local login to the remote login. If **Windows Authentication** is used, the windows credentials will be used to connect to a remote SQL server. If **SQL Server Authentication** is used, then the local login credentials will be passed to remote SQL Server. Note, to establish connection to remote server successfully, then the user with the exact same credentials must exist on the remote server otherwise when execute a linked server query:

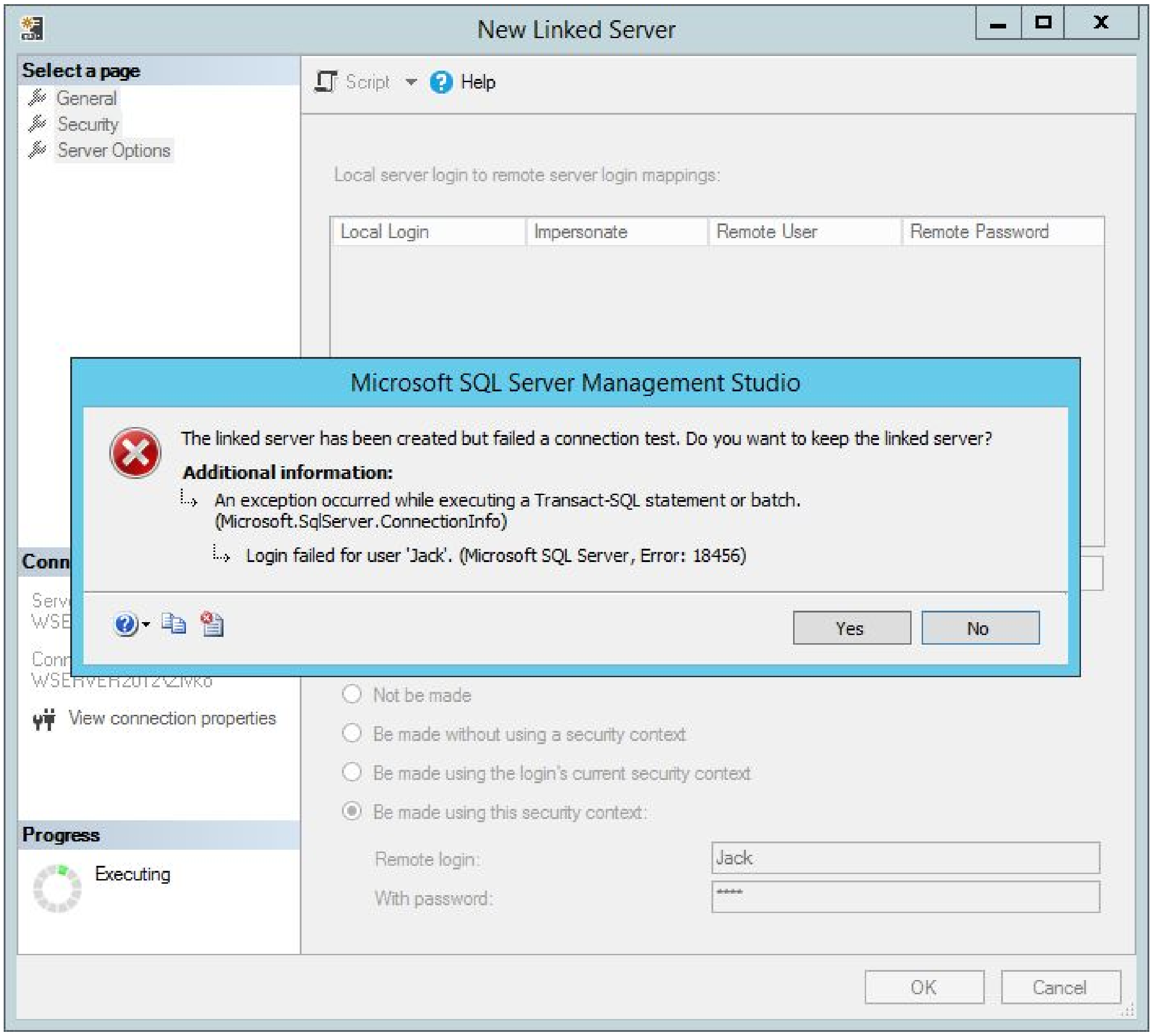
```
ELECT * FROM
WSERVER2012\SQLEXPRESS].AdventureWorks2014.HumanResources.Employee
```

The following message will appear:

```
Msg 18456, Level 14, State 1, Line 1
Login failed for user ‘Ben’.
```

Be made using this security context

The fourth option under the **For a login not defined in the list above, connections will** section is **Be made using this security context**. In the **Remote login** and **With password** fields, enter the credentials of the SQL Server Authentication login that exist on a remote server, otherwise the following error may occur:


New Linked Server
Select a page
General
Security
Server Options
Script
Help
Local server login to remote server login mappings:
Local Login
Impersonate
Remote User
Remote Password
Conn
Progress
Executing
View connection properties
Not be made
Be made without using a security context
Be made using the login's current security context
Be made using this security context:
Remote login:
With password:
OK
Cancel
Yes
No

Microsoft SQL Server Management Studio

The linked server has been created but failed a connection test. Do you want to keep the linked server?

Additional information:

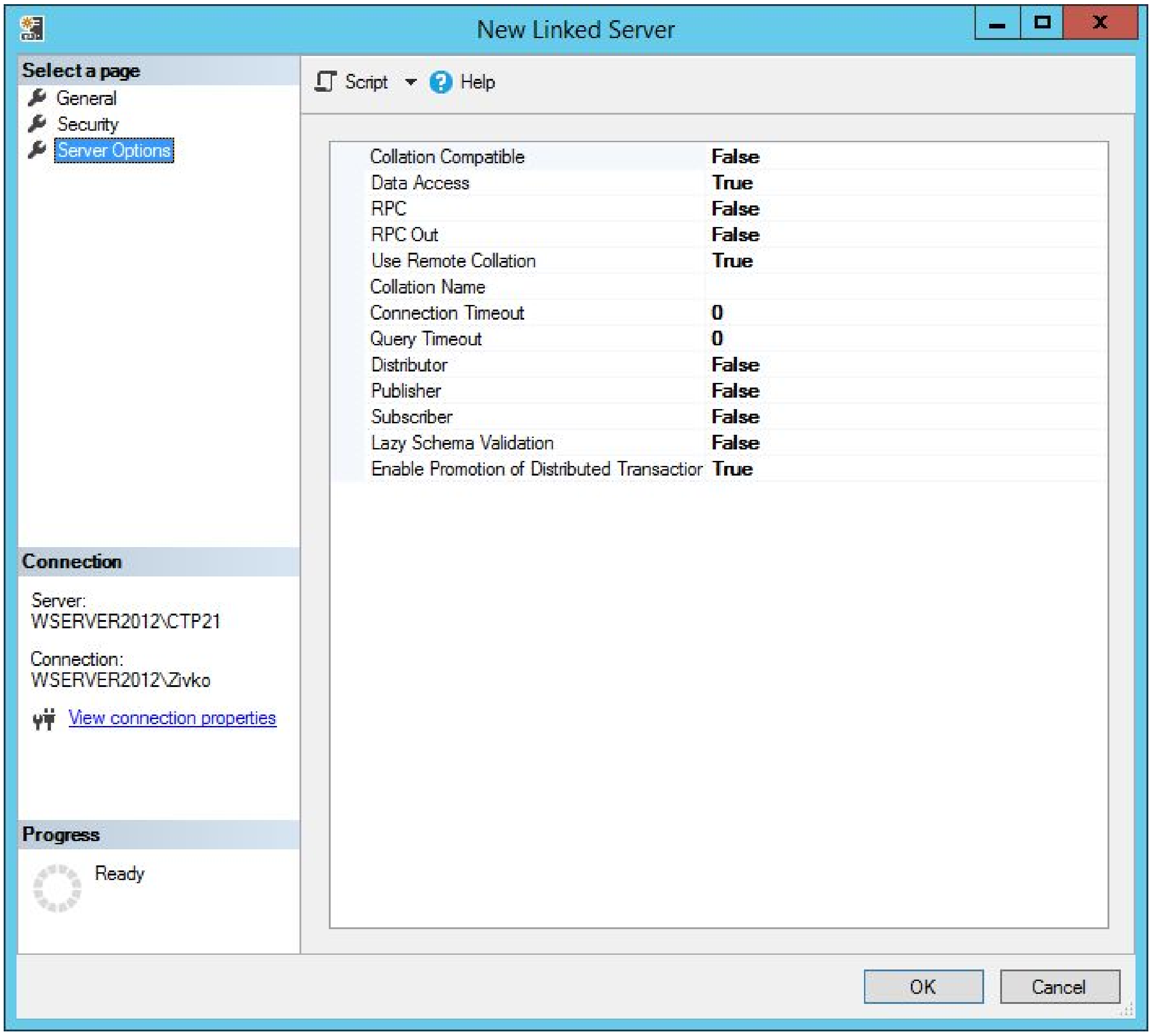
An exception occurred while executing a Transact-SQL statement or batch. (Microsoft.SqlServer.ConnectionInfo)

Login failed for user 'Jack'. (Microsoft SQL Server, Error: 18456)

Remote login: Jack

With password: ****

The last item under the **Select a page** menu is the **Server Options** item. When selecting this option, the following window will be shown:


New Linked Server

Select a page
- General
- Security
- Server Options

Connection

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

View connection properties

Progress

Ready

Collation Compatible	False
Data Access	True
RPC	False
RPC Out	False
Use Remote Collation	True
Collation Name	
Connection Timeout	0
Query Timeout	0
Distributor	False
Publisher	False
Subscriber	False
Lazy Schema Validation	False
Enable Promotion of Distributed Transactior	True

Here, additional options for linked server can be seen or set.

Collation Compatible

The first option is the Collation Compatible option. This option is used to identify if the linked server has the same collation as the local server. This option should set to True only if is known that the linked server has the same collation as the local, otherwise it should be set to **False** (default).

Data Access

This option is used to allow/deny access to the linked server data. If this option is set to **False,** the access to remote will be denied. This option is useful to disable access to a remote server temporally. The following message will appear when execute a linked server query and this option is set to False:

```
Msg 7411, Level 16, State 1, Line 1
Server 'WSERVER2012\SQLEXPRESS' is not configured for DATA ACCESS.
```

By default, the option is set to **True**

RPC and RCP Out

This RCP (Remote Procedure Call) is used to enable the access to remote procedures to be called from the linked server or to be called to the linked server.
If these options are set to **False,** the following error will appear when some procedures from the linked server are called:

```
Msg 7411, Level 16, State 1, Line 4
Server 'WSERVER2012\SQLEXPRESS' is not configured for RPC.
```

By default, the False value are set for the RPC and RCP Out options

Use Remote Collation

When this option is set to **True,** the collection of remote columns will be used and collection specified in the **Collation Name** filed will be used for data source that are not SQL Server data source, but if the option is set to **False** then the collation for the local server will be used. By default, is set to False.

Collation Name

If the **Use Remote Collation** filed set to True, this option is used to specify the collation name of the linked server for the data source that is not SQL Server data source. When chose a collation name, it must be a collation that SQL Server supports.

Connection Timeout

This option is used to set the maximum time the local server should wait for to get a connection to the linked server SQL Server instance. If 0 (zero) is set, then the server option remote login timeout is used. By default, 10 second is set for this option. Note, the default value for SQL Server 2008 is 20 seconds.

Query Timeout

This option is used to set how long, in seconds, a remote process can take before time is out. The default value is 600 second (10 minutes). To disable query timeout put 0 (zero) in this field and the query will wait until it is completed.

Distributor

In this option, it can be specified whether the linked server is participating in replication as a distribution Publisher.

The Distributor is a database instance, that acts as a store for replication specific data associated with one or more Publishers

Publisher

In this option, it can be set whether the linked server to be a replication publisher or not. If True, the linked server is a publisher. Otherwise, is not.

The Publisher is a database instance, that makes data available to other locations through replication.

Subscriber

In this option, it can be specified whether the linked server is a replication subscriber or not.

A Subscriber is a database instance, that receives replicated data.

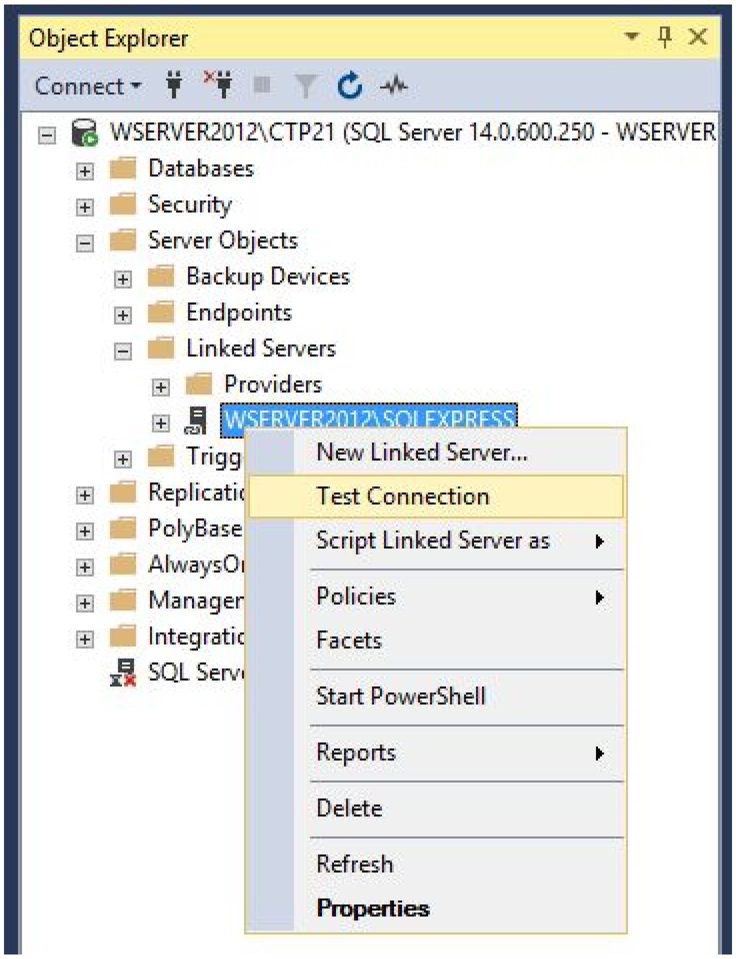
More information about **Distributor, Publisher, Subscriber** can be found on the Replication Publishing Model Overview page.

Lazy schema validation

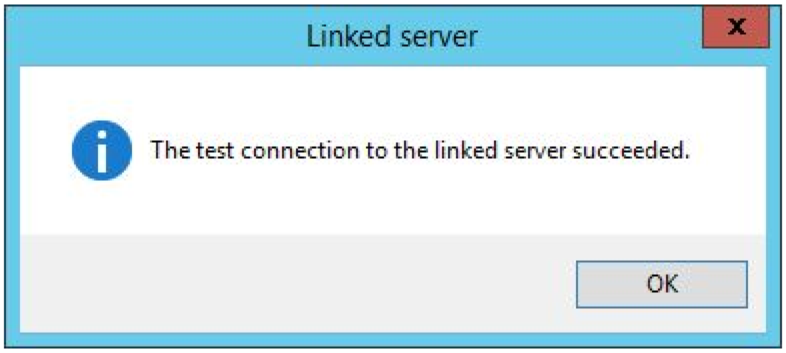
This option checks schema changes that have occurred since compilation in the remote tables. If this option is set to False (default state), SQL Server checks changes before the execution of a query and if there are some changes, it recompiles the query. If the Lazy schema validation is set to True, an SQL Server delay schema checks the remote tables until query execution.

Enable Promotion of Distributed Transactions

This option is used to protect the actions of a server-to-server procedure through a Microsoft Distributed Transaction Coordinator (MS DTC) transaction. If this option is set to True calling a remote stored procedure starts a distributed transaction and enlists the transaction with MS DTC.
Now, when everything is set, click the OK button on the New Linked Server dialog. A newly created linked server will appear under the Linked Server folder.
To test that linked server if it works properly, go right-clicking on that linked server and choose **Test Connection**:



If a connection with linked server is established successfully, the following info message box will appear:

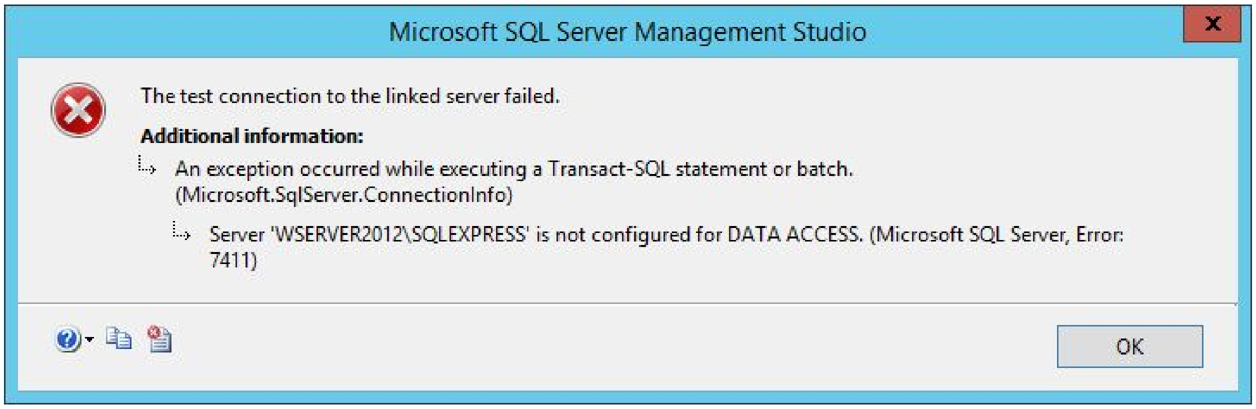


Linked server

The test connection to the linked server succeeded.

OK

Otherwise, an error message will be displayed that shows a problem that prevents connection to be successfully established:


Microsoft SQL Server Management Studio

The test connection to the linked server failed.

Additional information:

An exception occurred while executing a Transact-SQL statement or batch. (Microsoft.SqlServer.ConnectionInfo)

Server 'WSERVER2012\SQLEXPRESS' is not configured for DATA ACCESS. (Microsoft SQL Server, Error: 7411)

Querying data using a linked server

Querying data using the linked server is a little bit different then querying data from the local SQL Server. In the normal queries, usually, two part notation is used **[Schema].[ObjectName],** for example HumanResources.Employee:

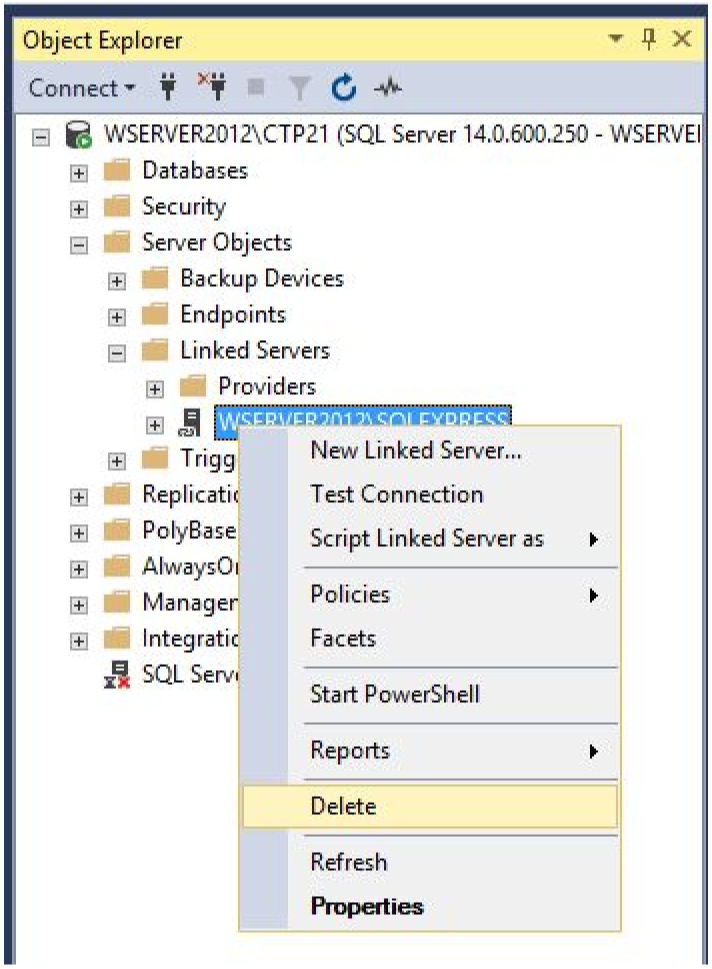
```
:LECT * FROM HumanResources.Employee e
```

When querying a table from a linked server, the fourth part notation is used **LinkedServer.Database.Schema.ObjectName**. To get data from the Employee table which is located in a database on the linked server, querying code will look like this:

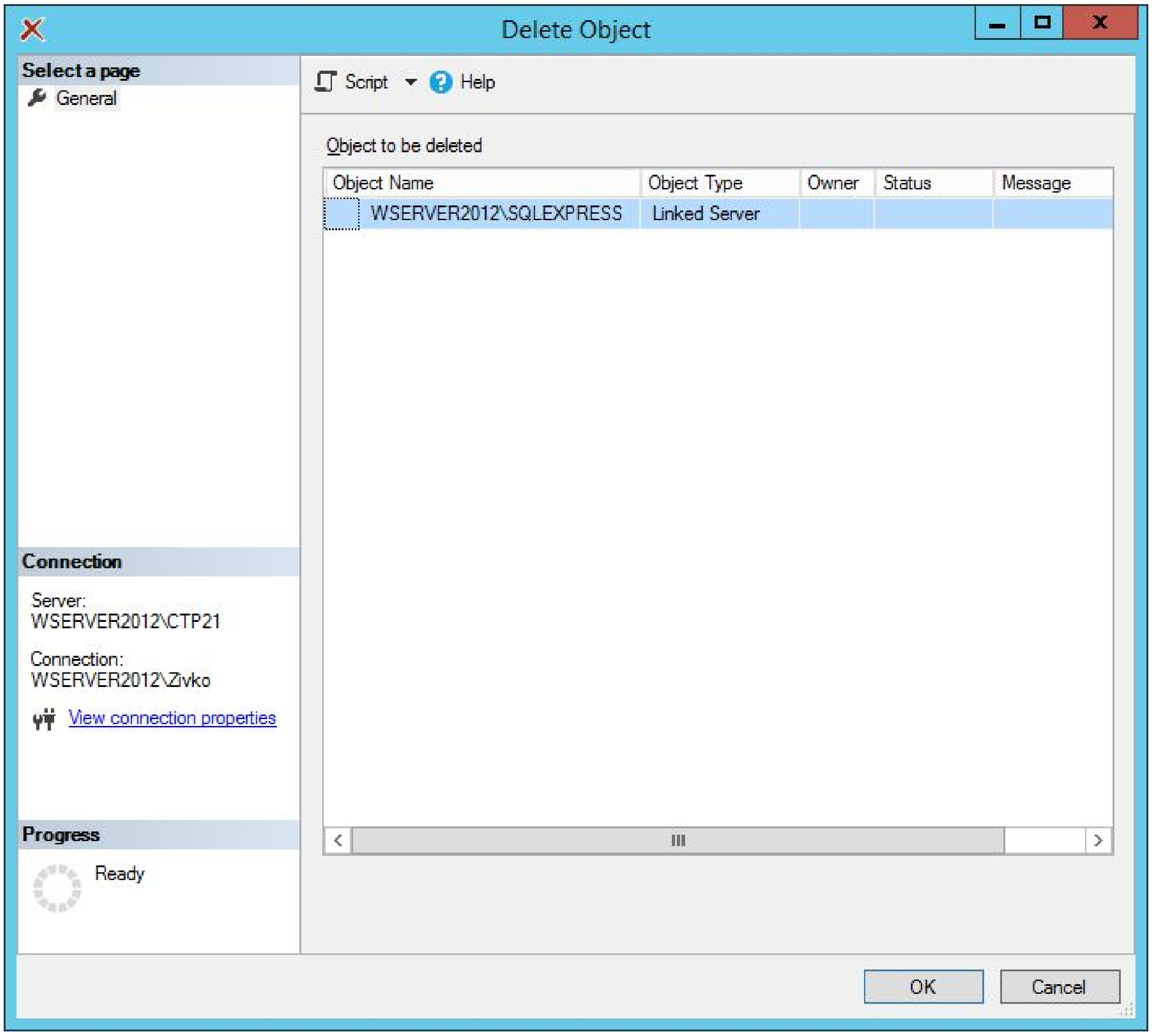
```
:LECT * FROM
:IVKO\SQLEXPRESS2016].[AdventureWorks2014].[HumanResources].[
nployee]
```

Deleting a linked server

To delete a linked server, under the Linked Servers folder, right click on the linked server and from the context menu choose the Delete command:



This will open the Delete Object dialog:



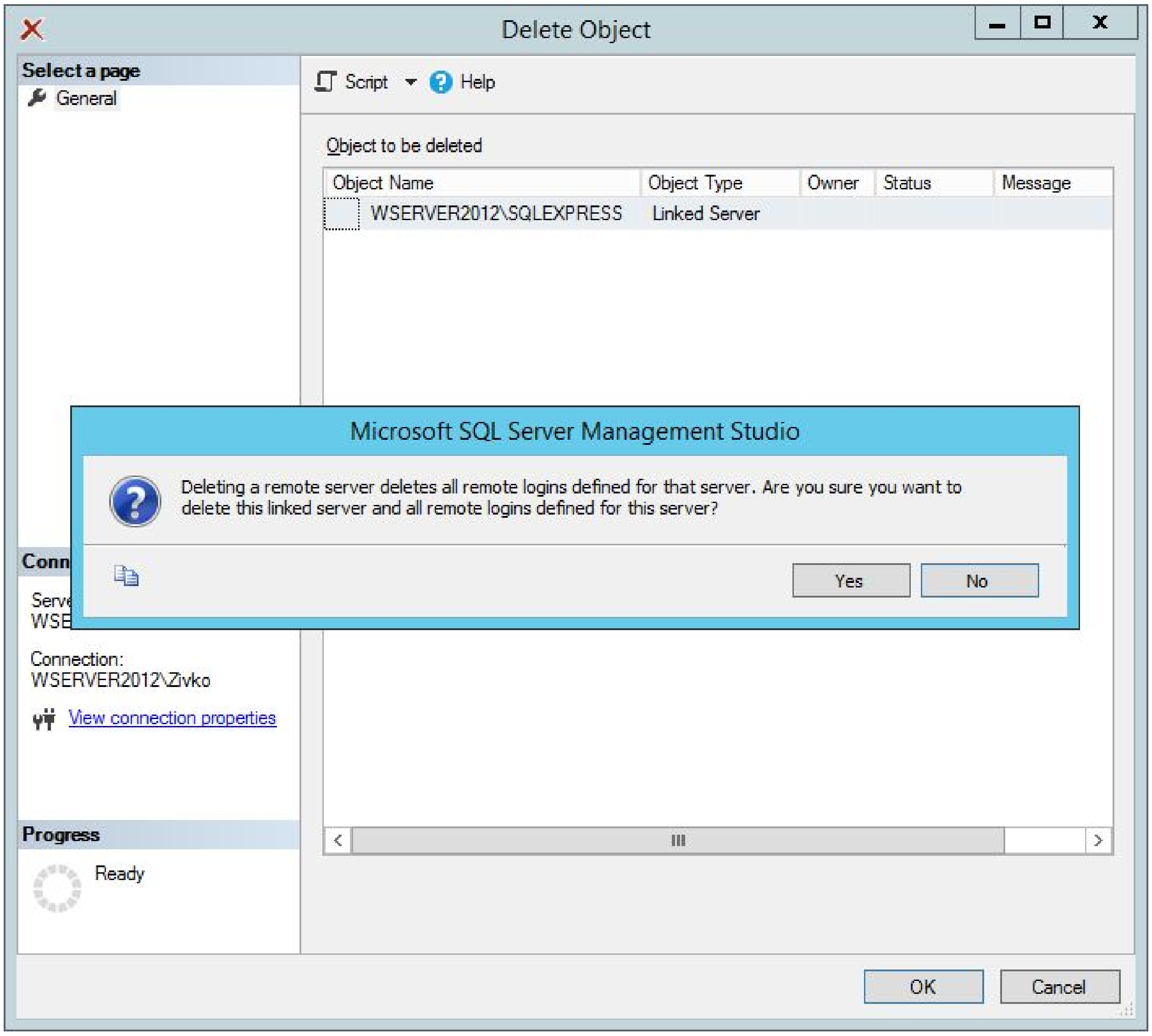
Object to be deleted

Object Name	Object Type	Owner	Status	Message
WSERVER2012\SQLEXPRESS	Linked Server			

Server:
WSERVER2012\CTP21

Connection:
WSERVER2012\Zivko

Click the **OK** button and from the message box, choose the **Yes** button:



Object to be deleted

Object Name	Object Type	Owner	Status	Message
WSERVER2012\SQLEXPRESS	Linked Server			

Connection:
WSERVER2012\Zivko

View connection properties

Microsoft SQL Server Management Studio

Deleting a remote server deletes all remote logins defined for that server. Are you sure you want to delete this linked server and all remote logins defined for this server?

If everything goes well the linked server will be removed from the Linked Servers folder.

BENEFITS OF LINKED SERVER:

1. Storage Reason
2. Legal Reason
3. Maintaince Reason
4. Null Values etc

SQL-SERVER NULL VALUES:

In databases a common issue is what value or placeholder do you use to represent a missing values. In SQL, this is solved with null. It is used to signify *missing* or *unknown* values. The keyword NULL is used to indicate these values. NULL really isn't a specific value as much as it is an indicator. Don't think of NULL as similar to zero or blank, it isn't the same. Zero (0) and blanks " ", are values.

In most of our beginning lessons we've assumed all tables contained data; however, SQL treats missing values differently. It is important to understand how missing values are used and their effect on queries and calculations.

SQL Server Database Growth and Autogrowth Settings:

DATABASE GROWTH AND AUTO GROWTH SETTING

SQL-SERVER CPU AFFINITY:

SQL Server Best Practices, Part I: Configuration:

Best Practice SQL Configuration

SQL Server Memory Consumption:

https://stackoverflow.com/questions/27010354/sql-server-memory-consumption

Understanding Transactions in SQL Server:

In this article, I'll cover the following:

- What a transaction is
- When to use transactions
- Understanding ACID properties
- Design of a Transaction
- Transaction state
- Specifying transaction boundaries
- T-SQL statements allowed in a transaction
- Local transactions in SQL Server 2012
- Distributed transactions in SQL Server 2012
- Guidelines to code efficient transactions
- How to code transactions

What Is a Transaction?

A transaction is a set of operations performed so all operations are guaranteed to succeed or fail as one unit.

Transaction is all or none:

A common example of a transaction is the process of transferring money from a checking account to a savings account.

This involves two operations:

1. Deducting money from the checking account and
2.
3. **Note**: in the USA a checking account is like a current account in India
4.
5. Adding it to the savings account.

Both must succeed together and the changes must be committed to the accounts, or both must fail together and rolled back so that the accounts are maintained in a consistent state. Under no circumstances should money be deducted from the checking account but not added to the savings account (or vice versa), you would at least not want this to happen with the transactions occurring with your bank accounts.

By using a transaction concept, both the operations, namely debit and credit, can be guaranteed to succeed or fail together. So both accounts remain in a consistent state all the time.

When to Use Transactions:

You should use transactions when several operations must succeed or fail as a unit. The following are some frequent scenarios where use of transactions is recommended:

- In batch processing, where multiple rows must be inserted, updated, or deleted as a single unit
- Whenever a change to one table requires that other tables be kept consistent
- When modifying data in two or more databases concurrently
- In distributed transactions, where data is manipulated in databases on various servers

When you use transactions, you put locks on data that is pending for permanent change to the database. No other operations can take place on locked data until the acquired lock is released. You could lock anything from a single row up to the entire database. This is called concurrency, which means how the database handles multiple updates at one time.

In the bank example above, locks will ensure that two separate transactions don't access the same accounts at the same time. If they do then either deposits or withdrawals could be lost.

Note: it's important to keep transactions pending for the shortest period of time. A lock stops others from accessing the locked database resource. Too many locks, or locks on frequently accessed resources, can seriously degrade performance.

Atomicity: A transaction is atomic if it's regarded as a single action rather than a collection of separate operations. So, only when all the separate operations succeed does a transaction succeed and is committed to the database. On the other hand, if a single operation fails during the transaction then everything is considered to have failed and must be undone (rolled back) if it has already taken place. In the case of the order-entry system of the Northwind database, when you enter an order into the Orders and Order Details tables, data will be saved together in both tables, or it won't be saved at all.

Consistency: The transaction should leave the database in a consistent state, whether or not it completed successfully. The data modified by the transaction must comply with all the constraints placed on the columns in order to maintain data integrity. In the case of Northwind, you can't have rows in the Order Details table without a corresponding row in the Orders table, since this would leave the data in an inconsistent state.

Isolation: Every transaction has a well-defined boundary; that is, it is isolated from another transaction. One transaction shouldn't affect other transactions running at the same time. Data modifications made by one transaction must be isolated from the data modifications made by all other transactions. A transaction sees data in the state it was in before another concurrent transaction modified it, or it sees the data after the second transaction has completed, but it doesn't see an intermediate state.

Durability: Data modifications that occur within a successful transaction are kept permanently within the system regardless of what else occurs. Transaction logs are maintained so that should a failure occur the database can be restored to its original state before the failure. As each transaction is completed, a row is entered in the database transaction log. If you have a major system failure that requires the database to be restored from a backup then you could then use this transaction log to insert (roll forward) any successful transactions that have taken place.

Every database software that offers support for transactions enforces these four ACID properties automatically.

Design of a Transaction:

Transactions represent real-world events such as bank transactions, airline reservations, remittance of funds, and so forth.

The purpose of transaction design is to define and document the high-level characteristics of transactions required on the database system, including the following:

- Data to be used by the transaction
- Functional characteristics of the transaction
- Output of the transaction
- Importance to users
- Expected rate of usage

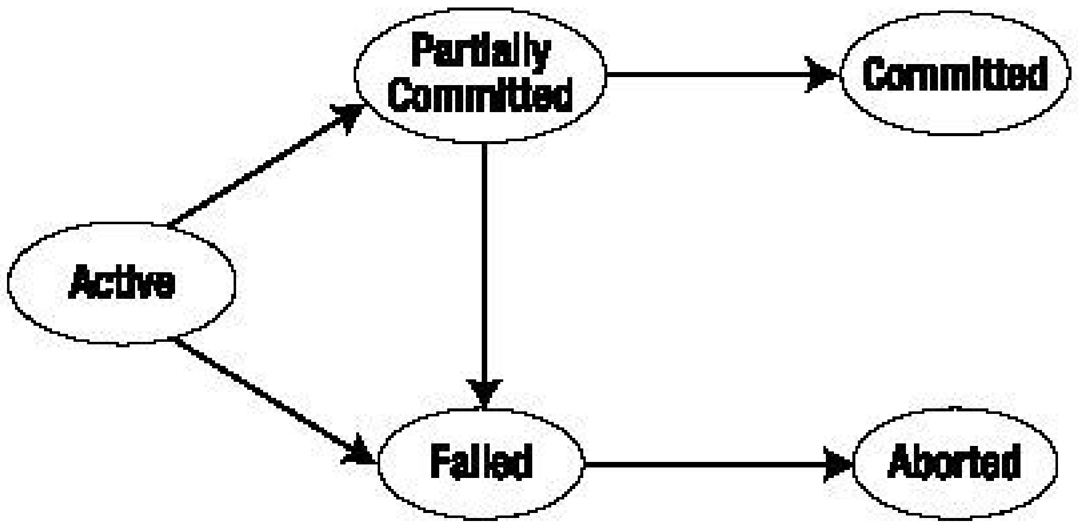
The following are the three main types of transactions

- Retrieval transactions: Retrieves data from the database to be displayed on the screen.
- Update transactions: Inserts new records, deletes old records, or modifies existing records in the database.
- Mixed transactions: Involves both the retrieval and updating of data.

Transaction State

In the absence of failures, all transactions complete successfully. However, a transaction may not always complete its execution successfully. Such a transaction is termed aborted.

A transaction that completes its execution successfully is said to be committed. Figure 1-1 shows that if a transaction has been partially committed then it will be committed but only if it has not failed and if the transaction has failed, it will be aborted.



```
Use AddressBook;



BEGIN TRANSACTION

-- Add a person

insert into person (personid, firstname, company)

values('0100', 'edee', 'pakistan')

Commit

Rollback



Select * from AddressBook..[Person]
```

Topics:

Clusterted Or Non Clusterted Index:

Clustered Index

- **Only one clustered index can be there in a table**
- **Sort the records and store them physically according to the order**
- **Data retrieval is faster than non-clustered indexes**
- **Do not need extra space to store logical structure**

Non Clustered Index

- **There can be any number of non-clustered indexes in a table**
- **Do not affect the physical order. Create a logical order for data rows and use pointers to physical data files**
- **Data insertion/update is faster than clustered index**
- **Use extra space to store logical str**

EXTRA TOPICS

1. ALARMS
2. MULTISERVER ADMINISTRATION
3. GOOD DATABASE HEALTH
4. DATABASE MAINTAINCE
   DMV
5. DMFS
6. PERFORMING MONITOR
7. DATABASE SNAPSHOT
8. SQL PROFILER

9. BACKUP RECOVERY PLAN