

# High Availability and Disaster Recovery

- ▶ The main point of a high-availability strategy is to keep the critical data as available as possible.
- ▶ The different high availability solutions provide ways to keep a database system online even at the time of hardware failure or other unforeseen incidents.
- ▶ Disaster recovery efforts are all about restoring systems and data to a previous acceptable state in the event of partial or complete failure of computers due to natural or technical causes

# Recovery Point Objective(RPO)

- ▶ The RPO is the amount of data one can lose, measured by time.
- ▶ It is likely to be different from system to system and application to application.
- ▶ For instance, a critical system may have an RPO measured in minutes while a non-critical one may have it measured in days.

# Recovery Time Objective (RTO)

- ▶ How much time is needed to bring systems online incase a disaster happened.
- ▶ DBAs need to work closely with other members of the IT team to establish the RPO and RTO

# High availability and Disaster recovery Solutions

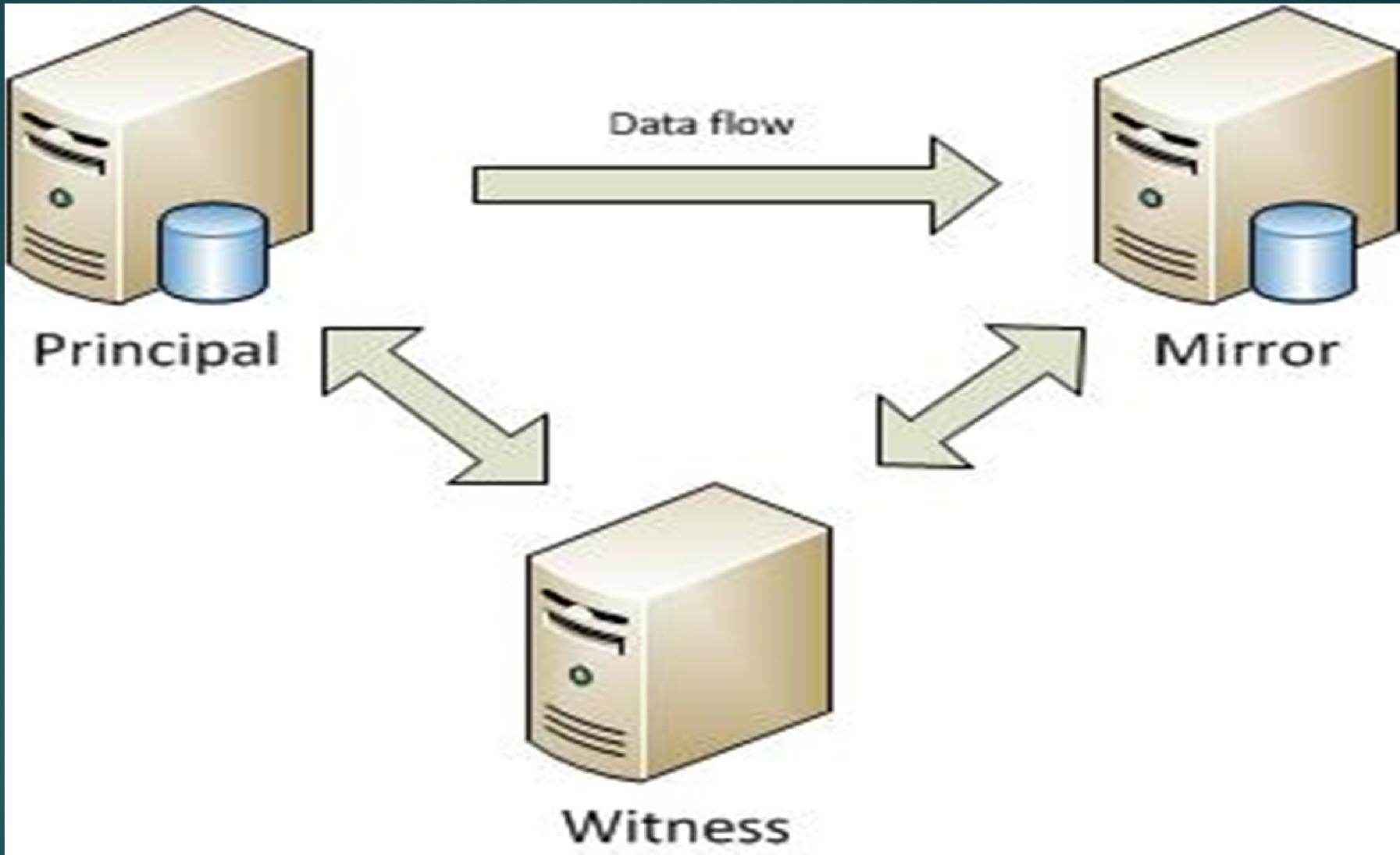
- ▶ Mirroring
- ▶ Log Shipping
- ▶ Replication
- ▶ AlwaysOn availability

# Mirroring

- SQL Server database mirroring is a disaster recovery and high availability technique that involves two SQL Server instances on the same or different machines. One SQL Server instance acts as a primary instance called the **principal**, while the other is a mirrored instance called the **mirror**. In special cases, there can be a third SQL Server instance that acts as a witness
  - 1. Principal Instance - Take a full backup and a log backup as well
  - 2. Copy the full/log backups from Principal Instance to Mirror instance
  - 3. Mirror Instance - Restore with NORECOVERY option the full backup
  - 4. Mirror Instance – Apply the log backup
  - 5. Principal Instance - Start synchronization

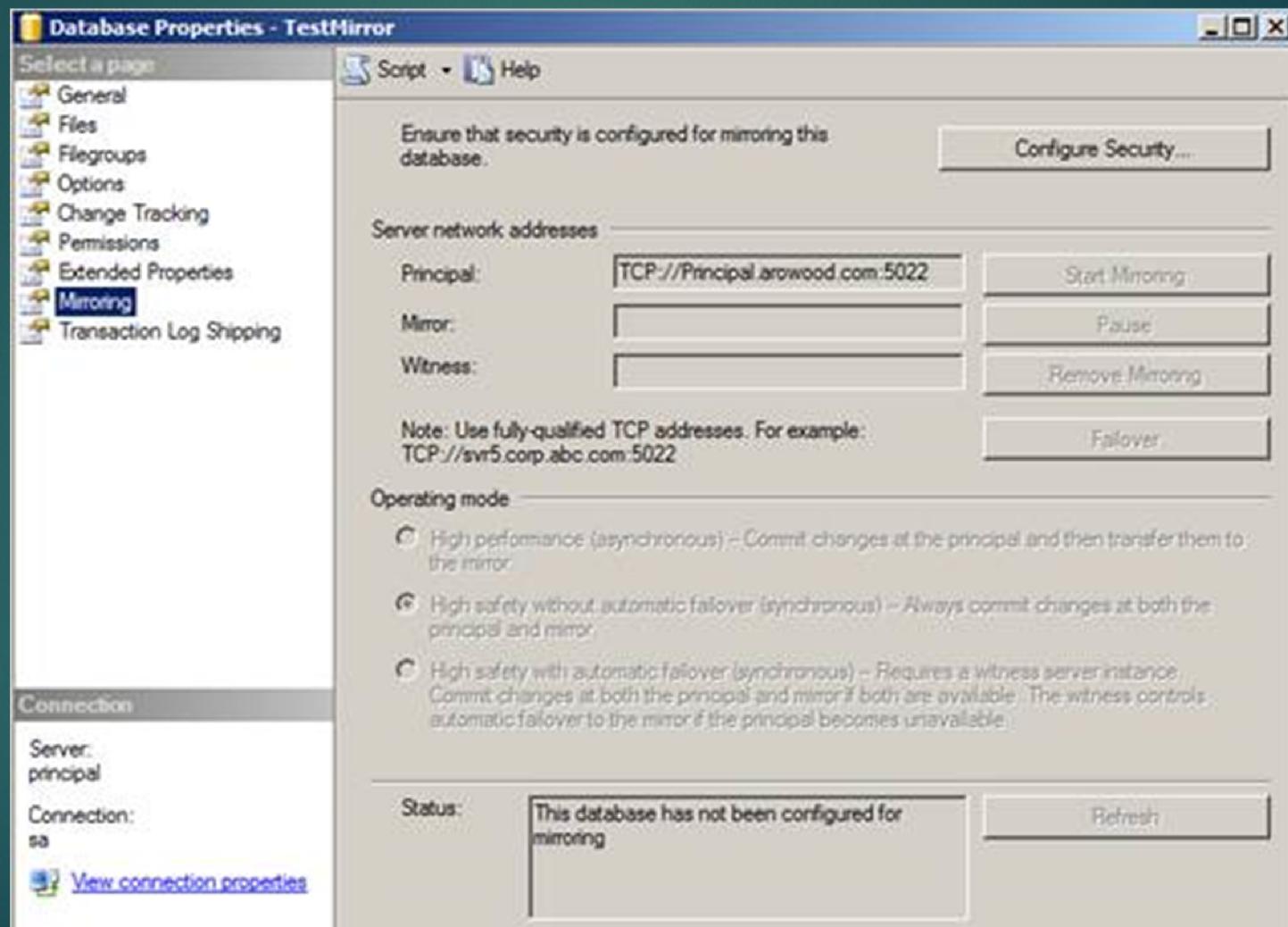
Witness: The role of the witness is to verify whether a given partner server is up and functioning. If the mirror server loses its connection to the principal server but the witness is still connected to the principal server, the mirror server does not initiate a failover

# Mirroring



# Mirroring

- From the Principal server, right click the database and choose "Tasks" | "Mirror" or choose "Properties" | "Mirroring".



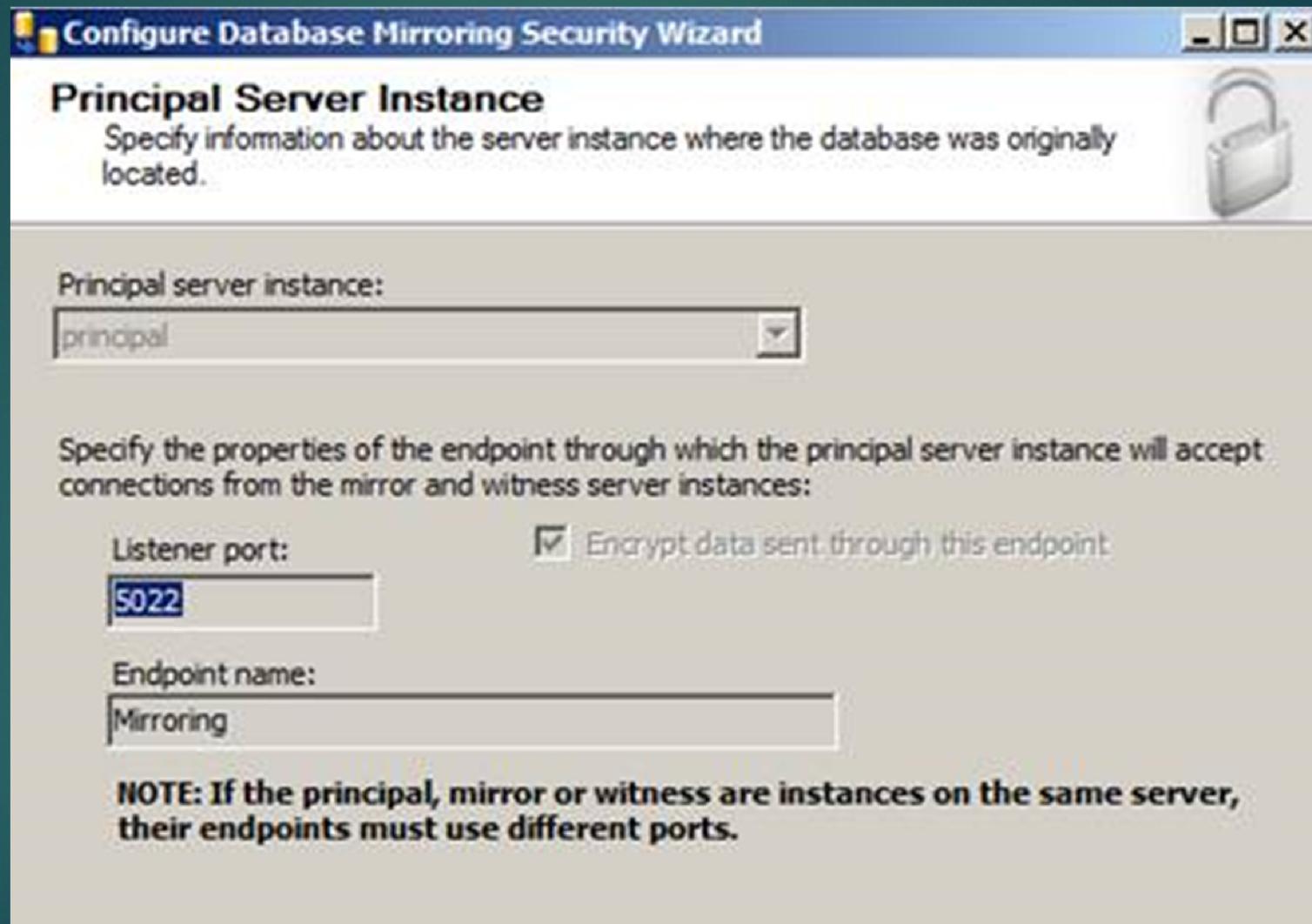
# Mirroring

- Click the "Configure Security" button and click "Next >" if the Configure Database Mirroring Security Wizard intro screen appears. The next screen should be the Include Witness Server screen:



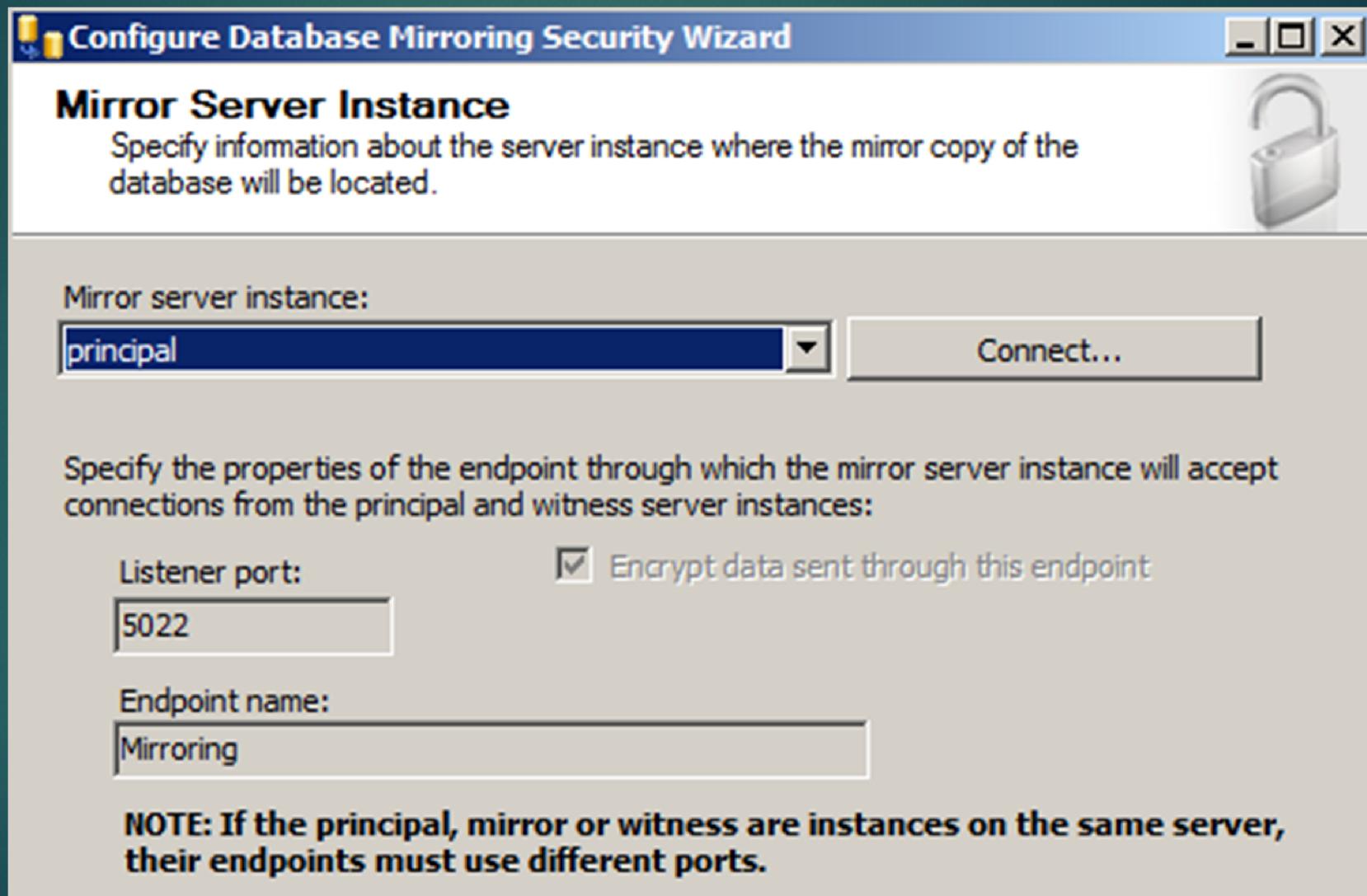
# Mirroring

- The next screen will give you options to configure the Principal Server Instance



# Mirroring

- The next screen will give you options to configure the Mirror Server Instance



# Mirroring

- To connect to the Mirror server instance we will need to click the "Connect..." button then select the mirror server and provide the correct credentials:



# Mirroring

- Once connected, we also notice our endpoint name is Mirroring and we are listening on port 5022.  
Click "Next >" and you'll see the Service Accounts screen.



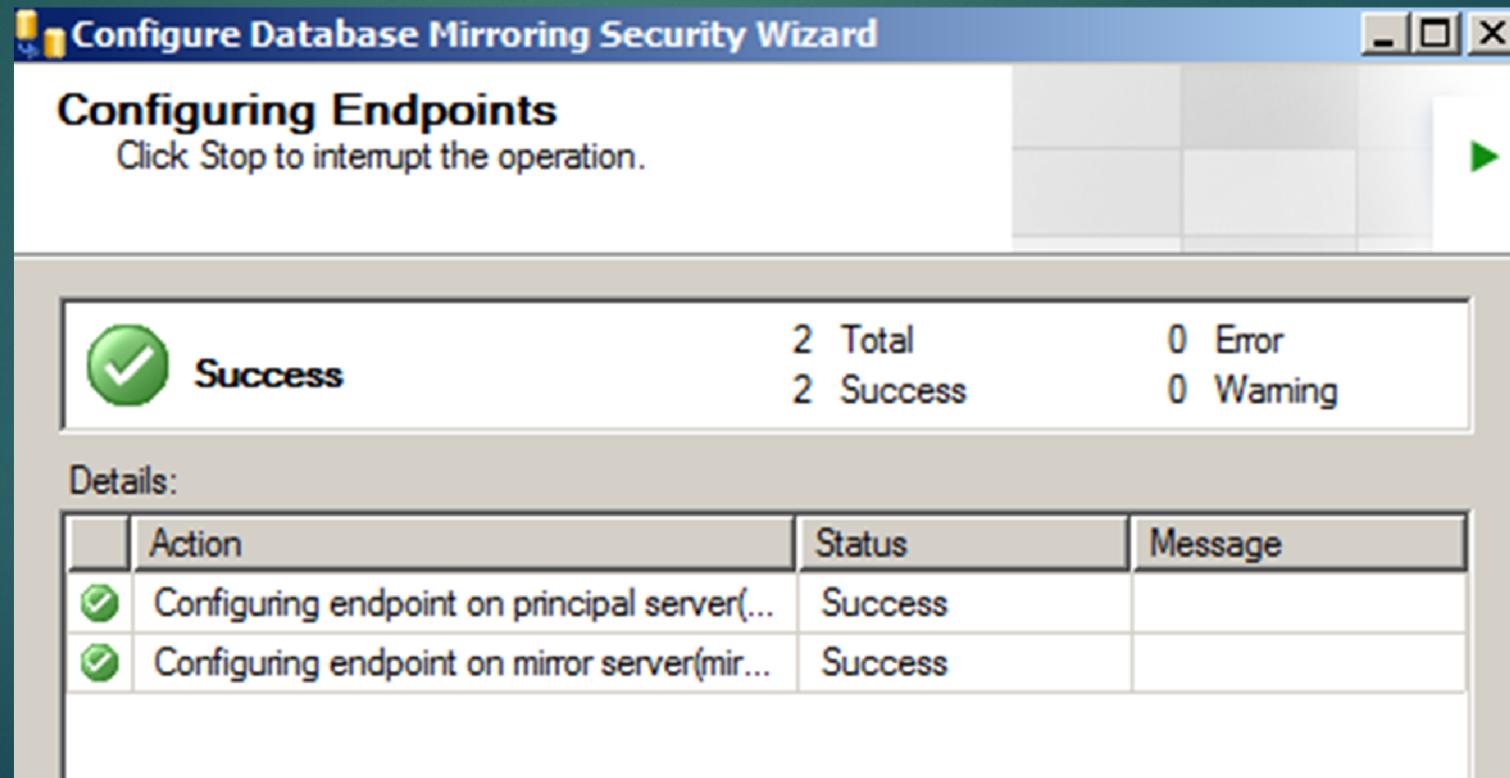
# Mirroring

When using Windows Authentication, if the server instances use different accounts, specify the service accounts for SQL Server. These service accounts must all be domain accounts (in the same or trusted domains).

If all the server instances use the same domain account or use certificate-based authentication, leave the fields blank.

Since my service accounts are using the same domain account, I'll leave this blank.

Click "Finish" and you'll see a Complete the Wizard screen that summarizes what we just configured. Click "Finish" one more time.



# Mirroring

- If you see the big green check mark that means Database Mirroring has been configured correctly. However, just because it is configured correctly doesn't mean that database mirroring is going to start...

Next screen that pops up should be the Start/Do Not Start Mirroring screen:

**Database Properties**

 Specified database mirroring configuration settings :

Principal network address: TCP://Principal .com:5022  
Mirror network address: TCP://MIRROR com:5022  
Witness network address: None  
Operating mode: High safety without automatic failover (synchronous)

To use the specified network addresses for mirroring this database, click Start Mirroring. To wait to start mirroring, click Do Not Start Mirroring; you can then start mirroring by clicking Start Mirroring on the Mirroring page of the Database Properties dialog box. Alternatively, you can exit the Database Properties dialog box without starting mirroring now, but you will need to configure the operating modes and server network addresses again before you can start mirroring.



# Log Shipping

- Log Shipping is a basic level SQL Server high-availability technology that is part of SQL Server. It is an automated backup/restore process that allows you to create another copy of your database for failover.
- Log shipping involves copying a database backup and subsequent transaction log backups from the primary (source) server and restoring the database and transaction log backups on one or more secondary (Stand By / Destination) servers. The Target Database is in a standby or no-recovery mode on the secondary server(s) which allows subsequent transaction logs to be backed up on the primary and shipped (or copied) to the secondary servers and then applied (restored) there.

## ➤ Permissions

To setup a log-shipping you must have sysadmin rights on the server.

## ➤ Minimum Requirements

SQL Server 2005 or later

Standard, Workgroup or Enterprise editions must be installed on all server instances involved in log shipping.

The servers involved in log shipping should have the same case sensitivity settings.

The database must use the full recovery or bulk-logged recovery model

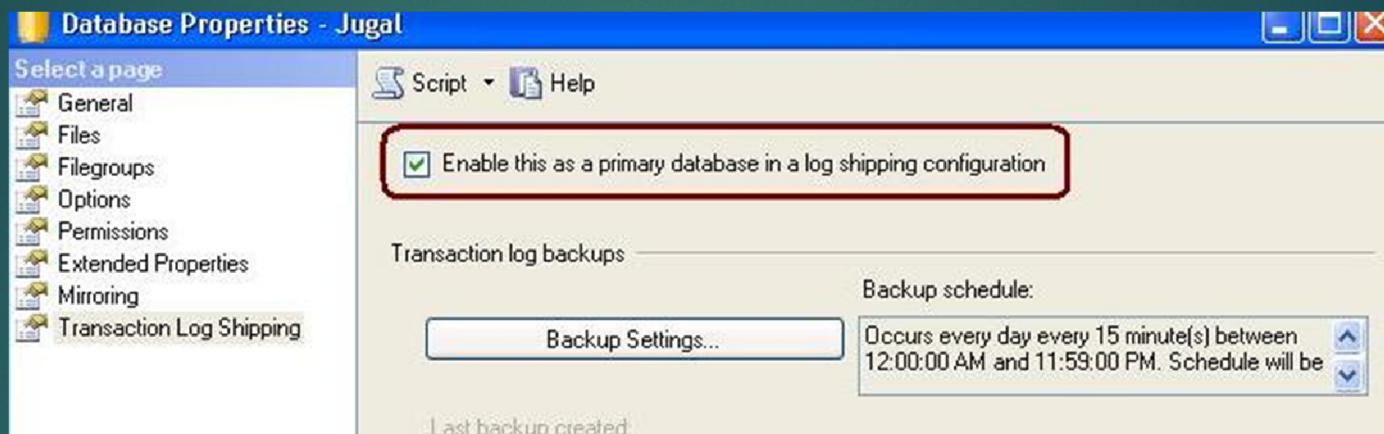
A shared folder for copying T-Log backup files

SQL Server Agent Service must be configured properly

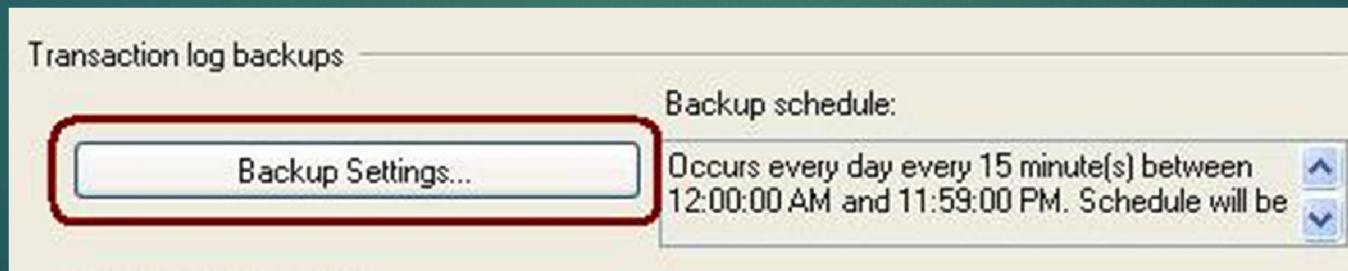
In addition, you should use the same version of SQL Server on both ends. It is possible to Log Ship from SQL 2005 to SQL 2008, but you can not do it the opposite way. Also, since Log Shipping will be primarily used for failover if you have the same versions on each end and there is a need to failover you at least know you are running the same version of SQL Server.

# Log Shipping

- On the primary server, right click on the database in SSMS and select Properties. Then select the **Transaction Log Shipping** Page. Check the "**Enable this as primary database in a log shipping configuration**" check box.

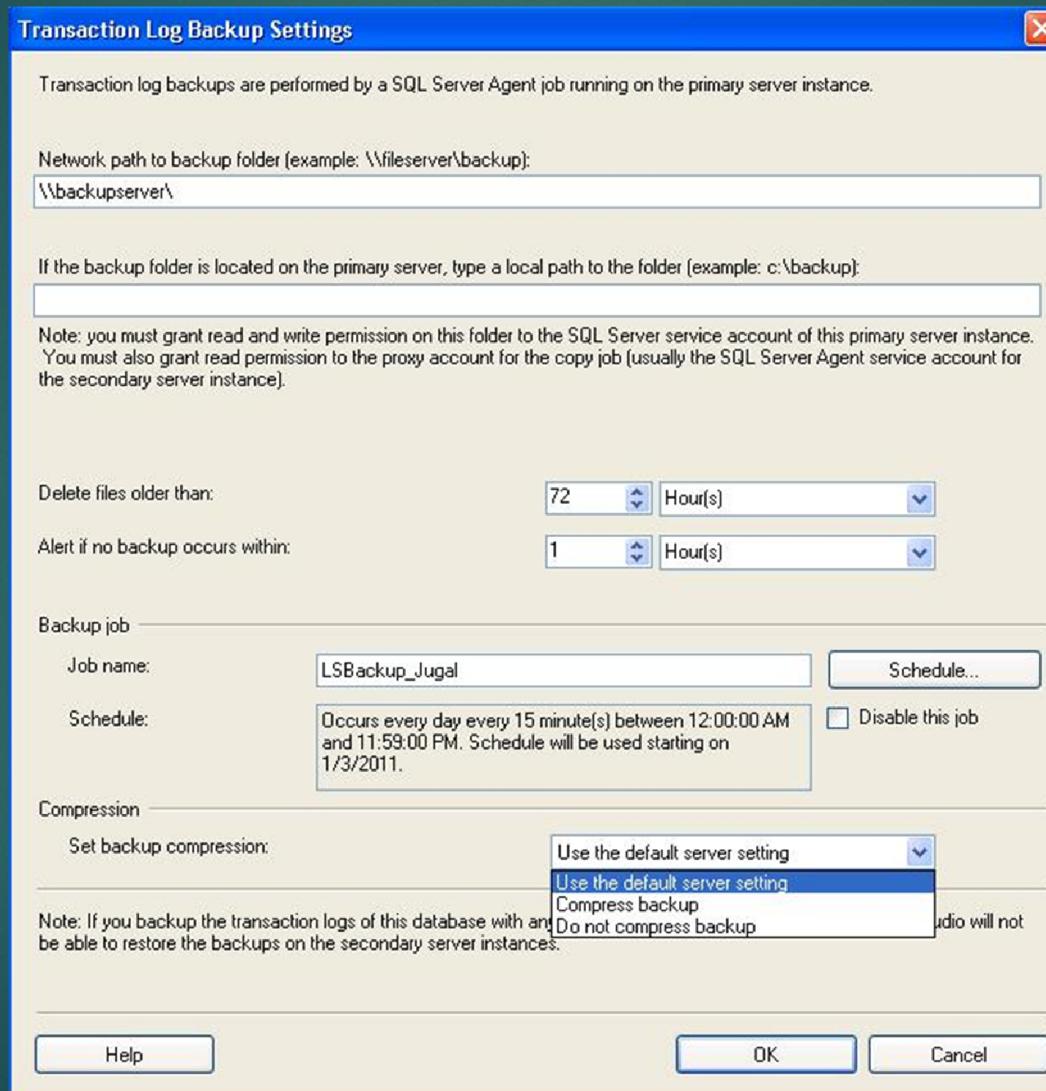


- The next step is to configure and schedule a transaction log backup. Click on Backup Settings... to do this.



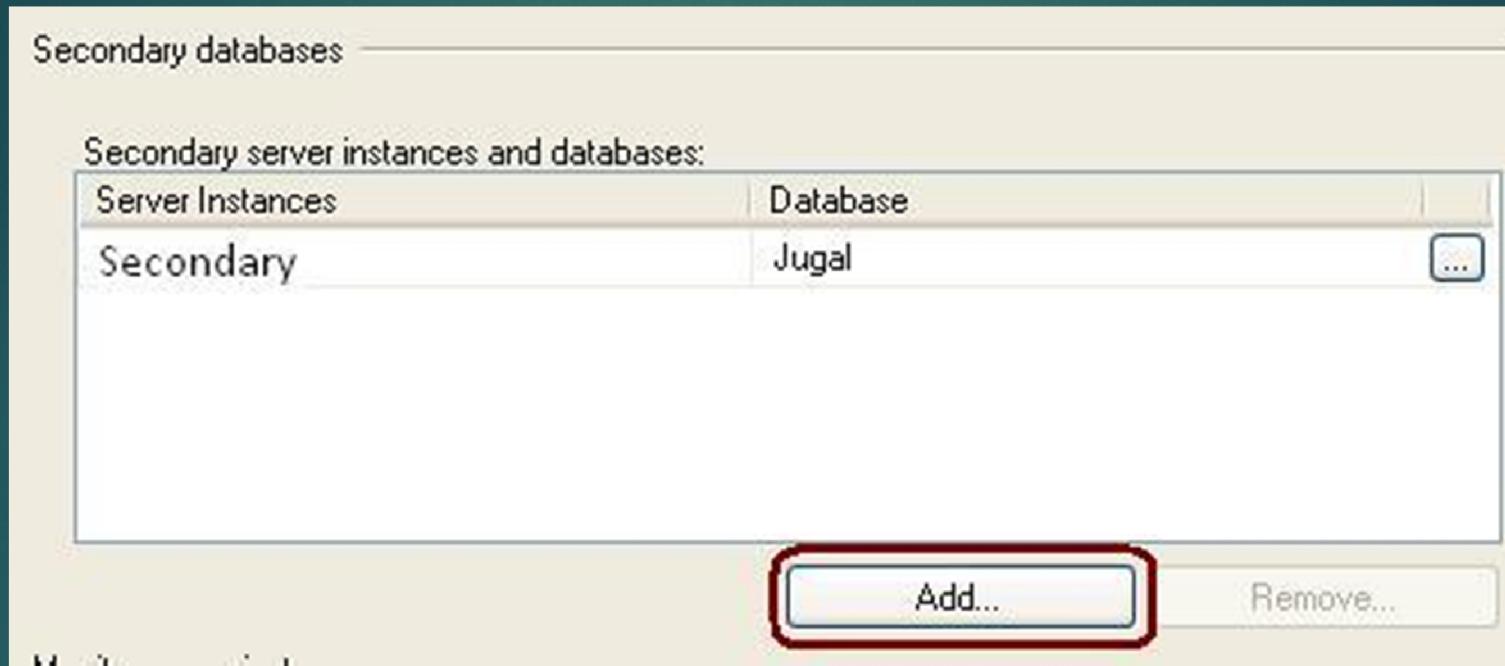
# Log Shipping

- If you are creating backups on a network share enter the network path or for the local machine you can specify the local folder path.



# Log Shipping

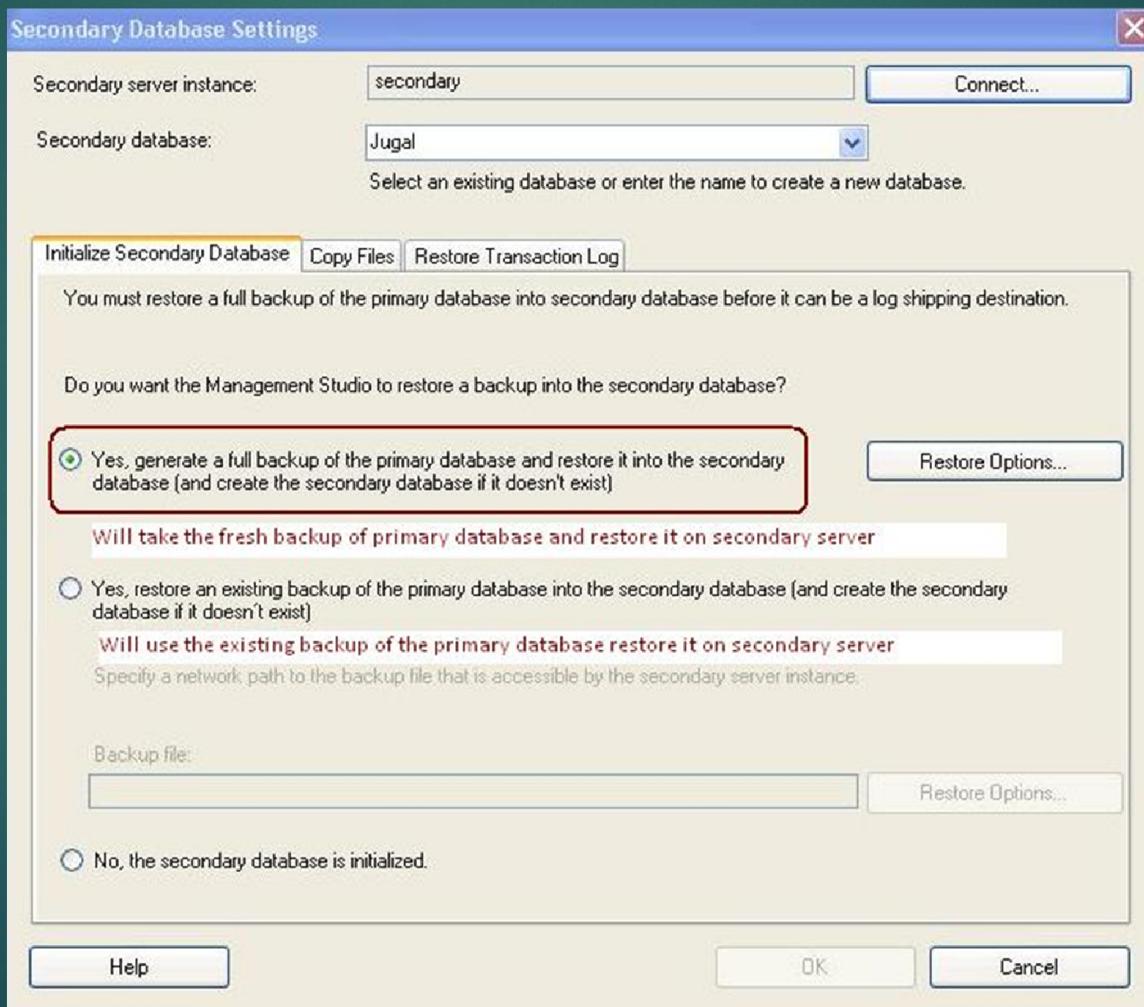
- In this step we will configure the secondary instance and database. Click on the **Add...** button to configure the Secondary Server instance and database. You can add multiple servers if you want to setup one to many server log-shipping.



- When you click the Add... button it will take you to the below screen where you have to configure the Secondary Server and database. Click on the Connect... button to connect to the secondary server. Once you connect to the secondary server you can access the three tabs as shown below.

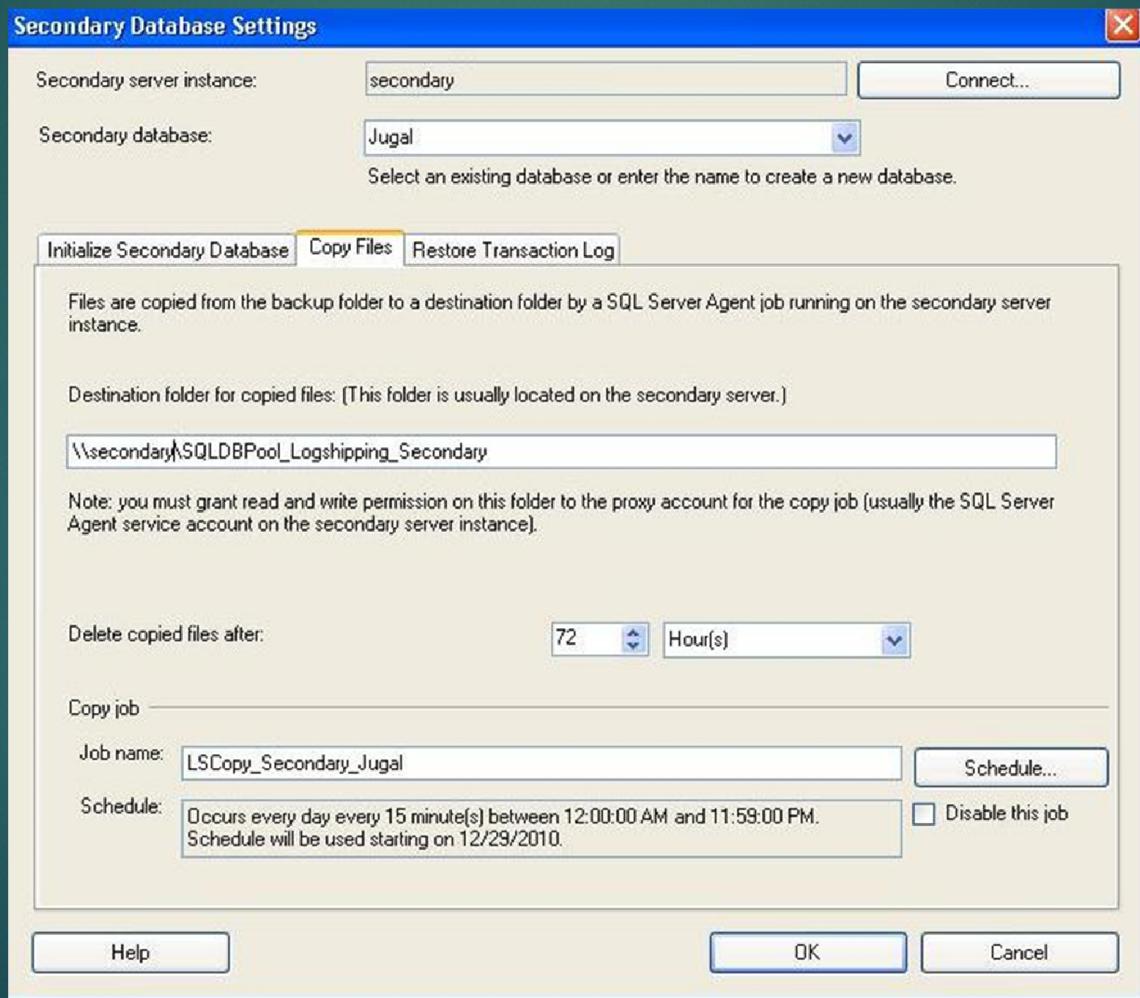
# Log Shipping

- In this step you can specify how to create the data on the secondary server. You have three options: create a backup and restore it, use an existing backup and restore or do nothing because you have manually restored the database and have put it into the correct state to receive additional backups



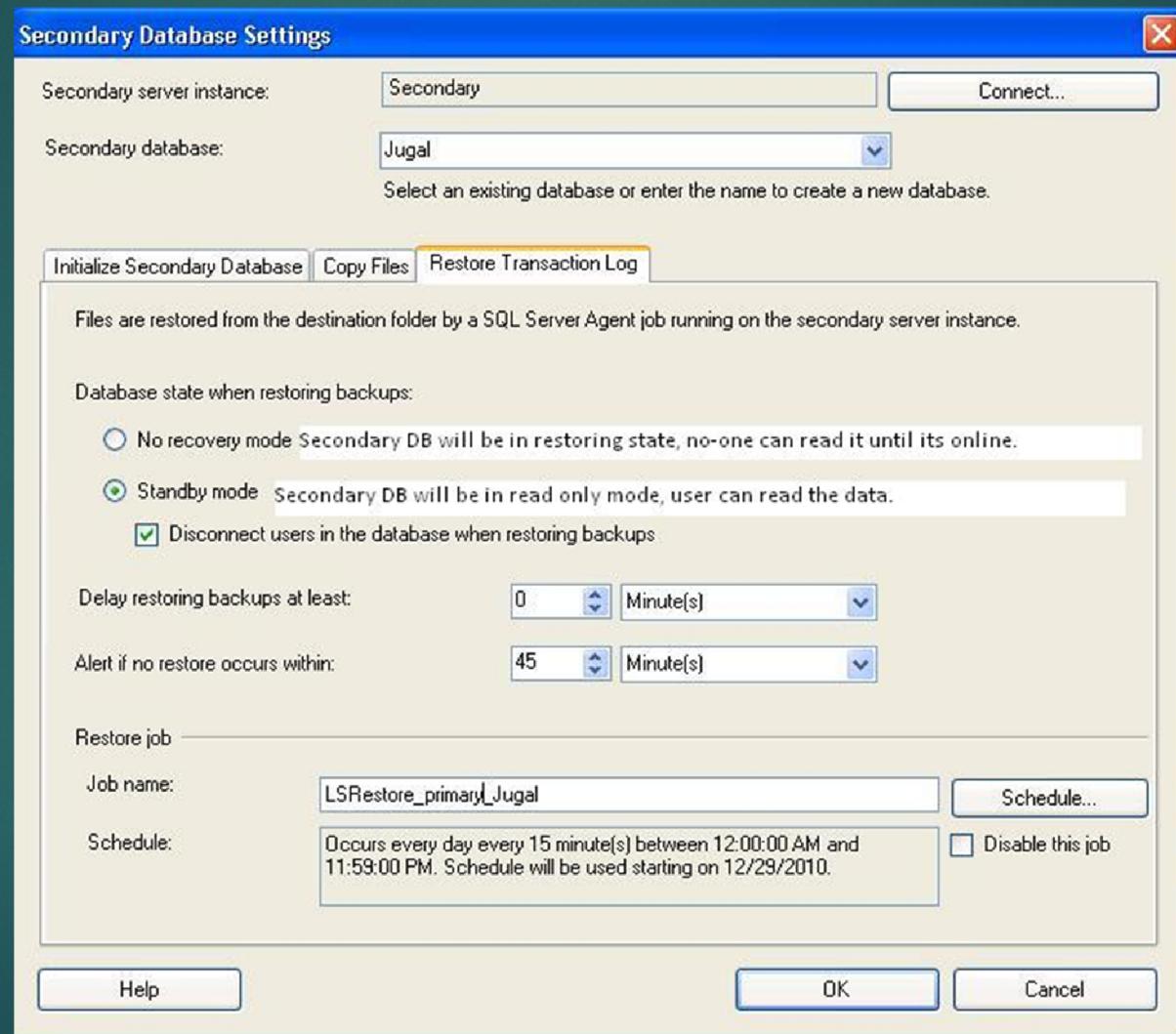
# Log Shipping

- In this step you can specify how to create the data on the secondary server. You have three options: create a backup and restore it, use an existing backup and restore or do nothing because you have manually restored the database and have put it into the correct state to receive additional backups



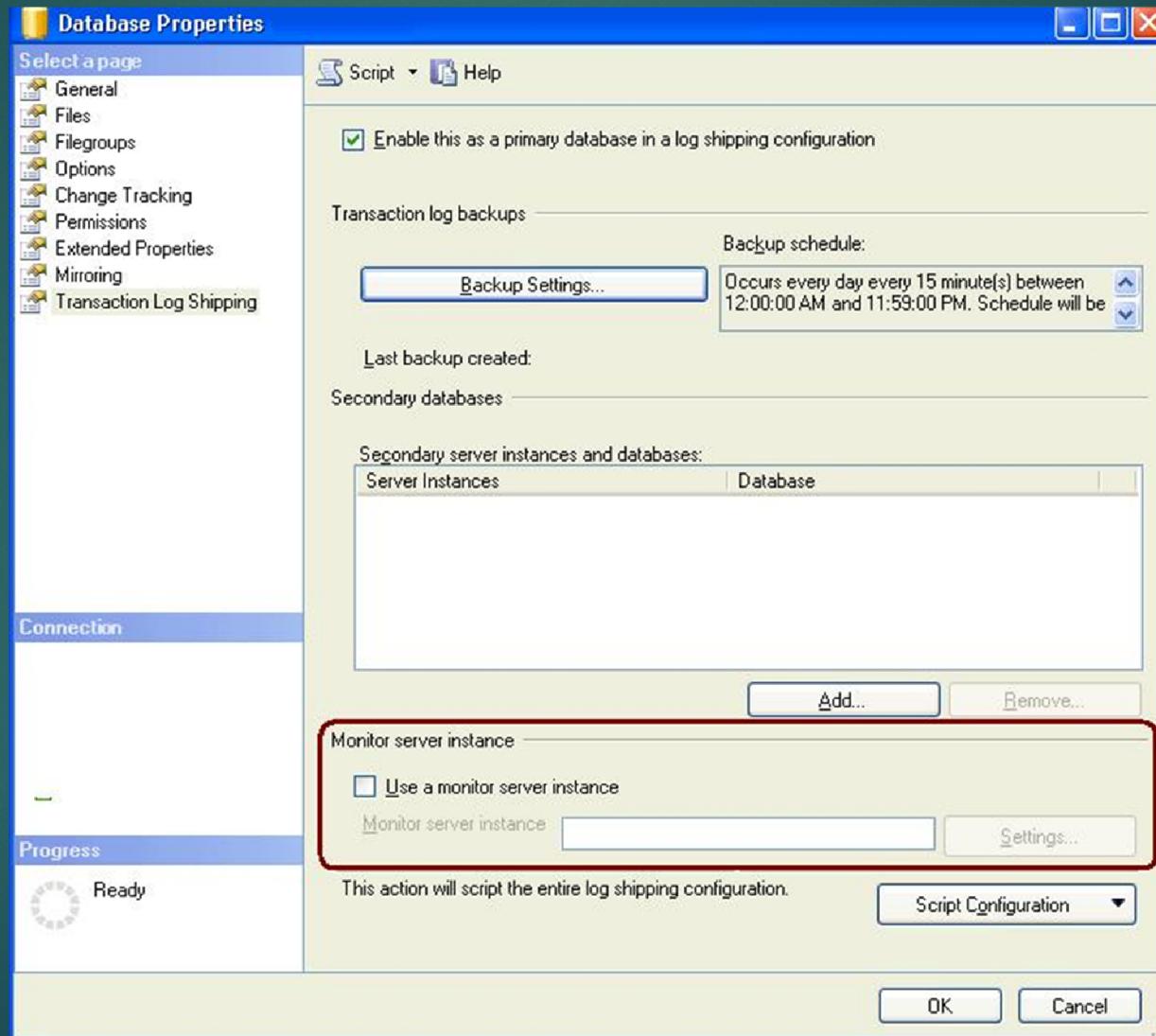
# Log Shipping

- Here you have to specify the database restoring state information and restore schedule. This will create the restore job on the secondary server.



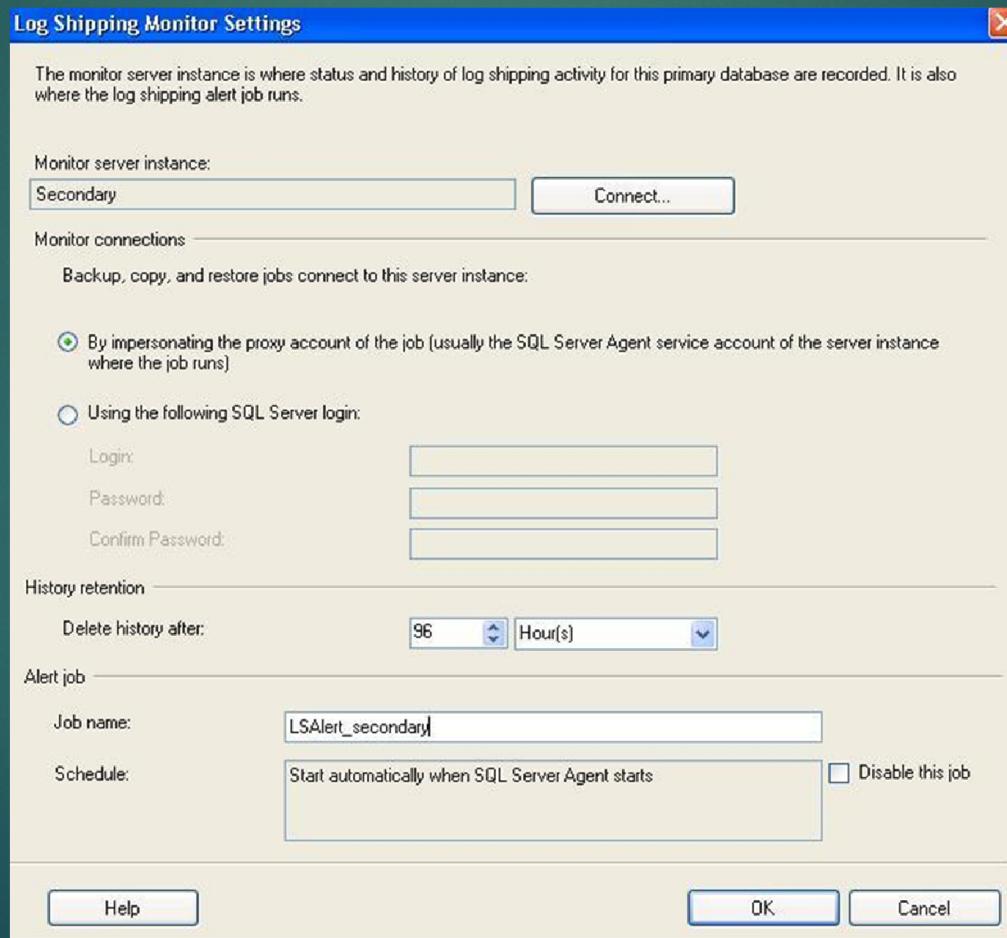
# Log Shipping

- In this step we will configure Log Shipping Monitoring which will notify us in case of any failure. Please note Log Shipping monitoring configuration is optional.



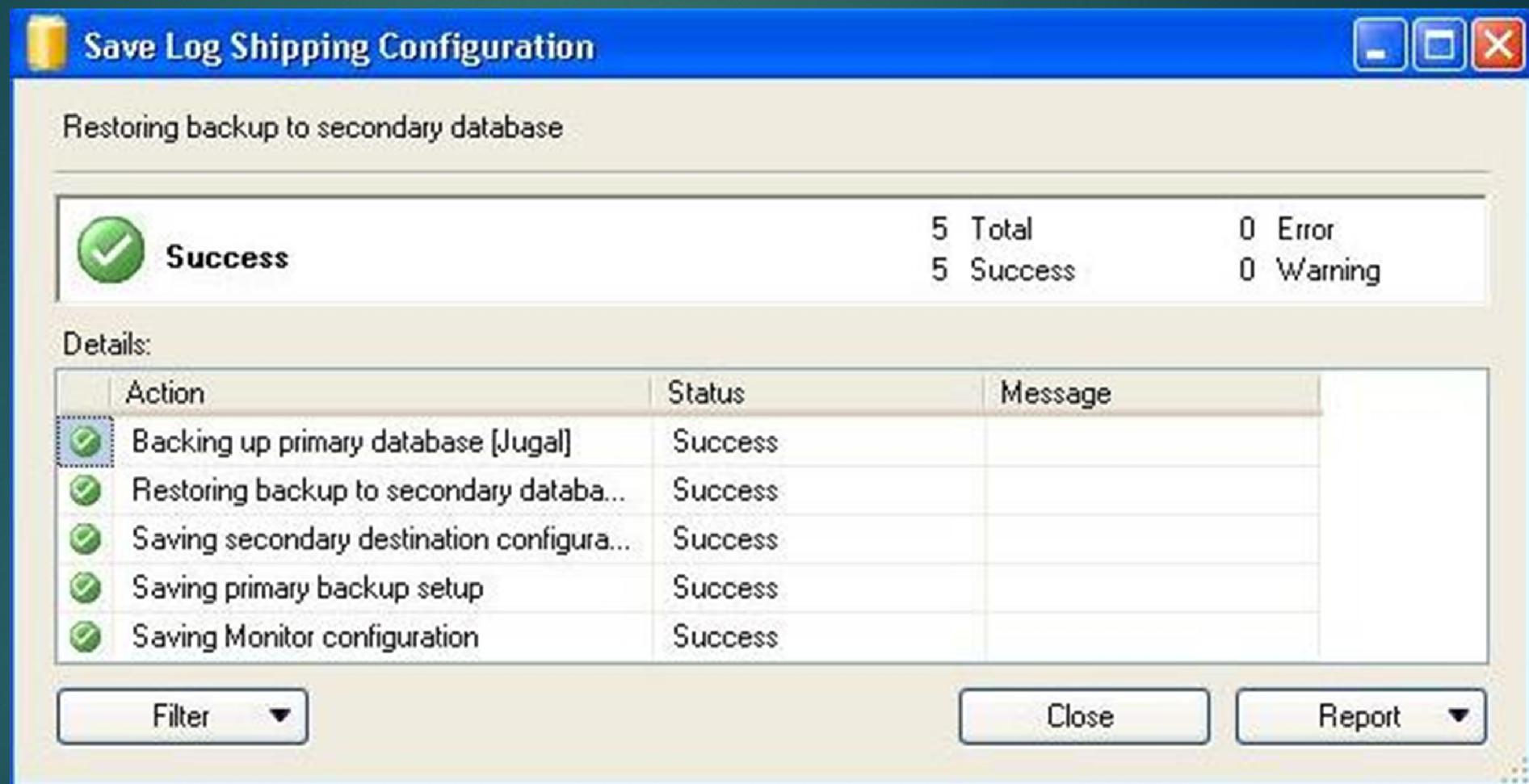
# Log Shipping

Click on **Settings...** button which will take you to the "**Log Shipping Monitor Settings**" screen. Click on **Connect ...** button to setup a monitor server. Monitoring can be done from the source server, target server or a separate SQL Server instance. We can configure alerts on source / destination server if respective jobs fail. Lastly we can also configure how long job history records are retained in the MSDB database. Please note that you cannot add a monitor instance once log shipping is configured.



# Log Shipping

- Click on the **OK** button to finish the Log Shipping configuration and it will show you the below screen.

A screenshot of a Windows application window titled "Save Log Shipping Configuration". The window has a blue header bar with the title and standard window controls (minimize, maximize, close). The main area is titled "Restoring backup to secondary database". It shows a summary table with a green checkmark icon and the word "Success". The summary table has three columns: "5 Total", "0 Error", "5 Success", and "0 Warning". Below this is a "Details:" section containing a table with five rows, each with a green checkmark icon and the word "Success". The table has three columns: "Action", "Status", and "Message". At the bottom are buttons for "Filter" (with a dropdown arrow), "Close", and "Report" (with a dropdown arrow).

Action	Status	Message
Backing up primary database [Jugal]	Success	
Restoring backup to secondary database	Success	
Saving secondary destination configuration	Success	
Saving primary backup setup	Success	
Saving Monitor configuration	Success	