**Database Files and Filegroups**

At a minimum, every SQL Server database has two operating system files: a data file and a log file. Data files contain data and objects such as tables, indexes, stored procedures, and views. Log files contain the information that is required to recover all transactions in the database. Data files can be grouped together in filegroups for allocation and administration purposes.

## Database Files

SQL Server databases have three types of files, as shown in the following table.

| **File** | **Description** |
| --- | --- |
| Primary | The primary data file contains the startup information for the database and points to the other files in the database. User data and objects can be stored in this file or in secondary data files. Every database has one primary data file. The recommended file name extension for primary data files is .mdf. |
| Secondary | Secondary data files are optional, are user-defined, and store user data. Secondary files can be used to spread data across multiple disks by putting each file on a different disk drive. Additionally, if a database exceeds the maximum size for a single Windows file, you can use secondary data files so the database can continue to grow.  The recommended file name extension for secondary data files is .ndf. |
| Transaction Log | The transaction log files hold the log information that is used to recover the database. There must be at least one log file for each database. The recommended file name extension for transaction logs is .ldf. |

For example, a simple database named **Sales** can be created that includes one primary file that contains all data and objects and a log file that contains the transaction log information. Alternatively, a more complex database named **Orders** can be created that includes one primary file and five secondary files. The data and objects within the database spread across all six files, and the four log files contain the transaction log information.

By default, the data and transaction logs are put on the same drive and path. This is done to handle single-disk systems. However, this may not be optimal for production environments. We recommend that you put data and log files on separate disks.

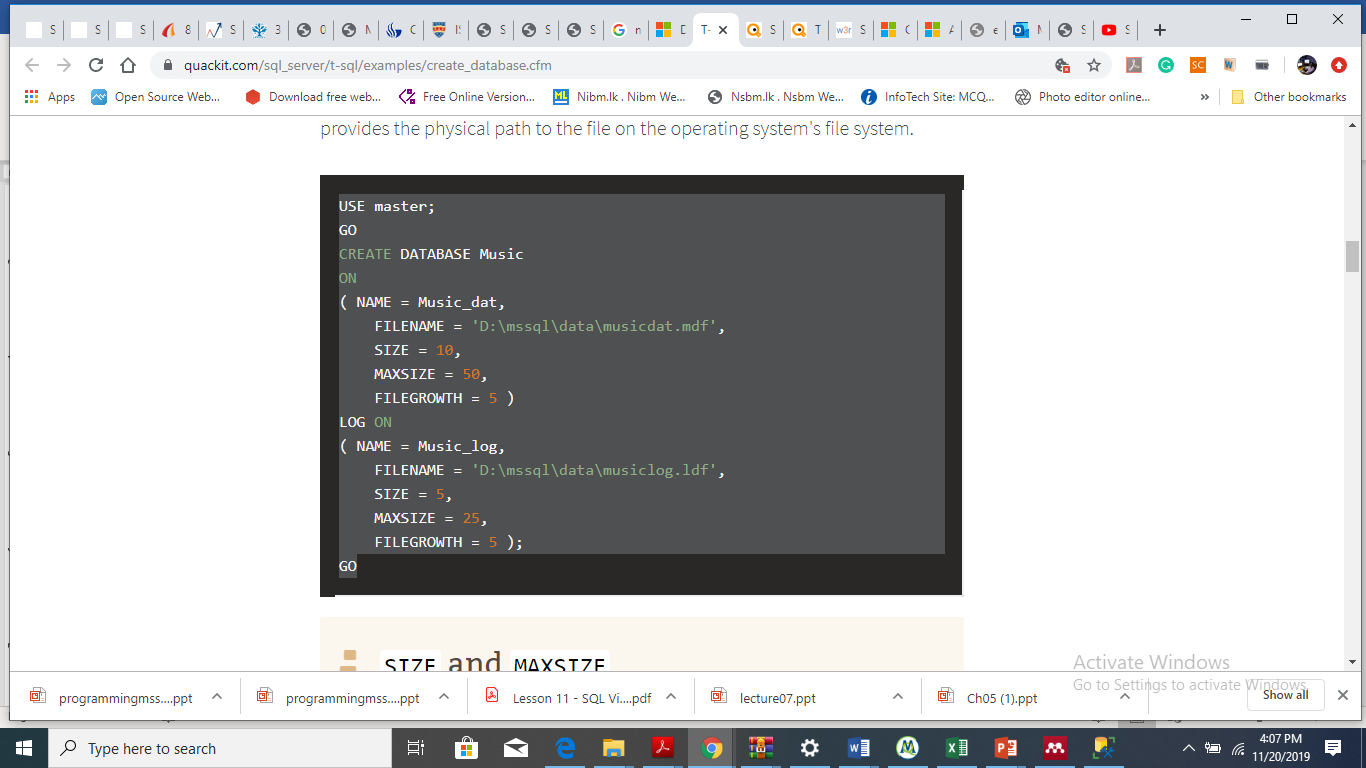
### Logical and Physical File Names

SQL Server files have two file name types:

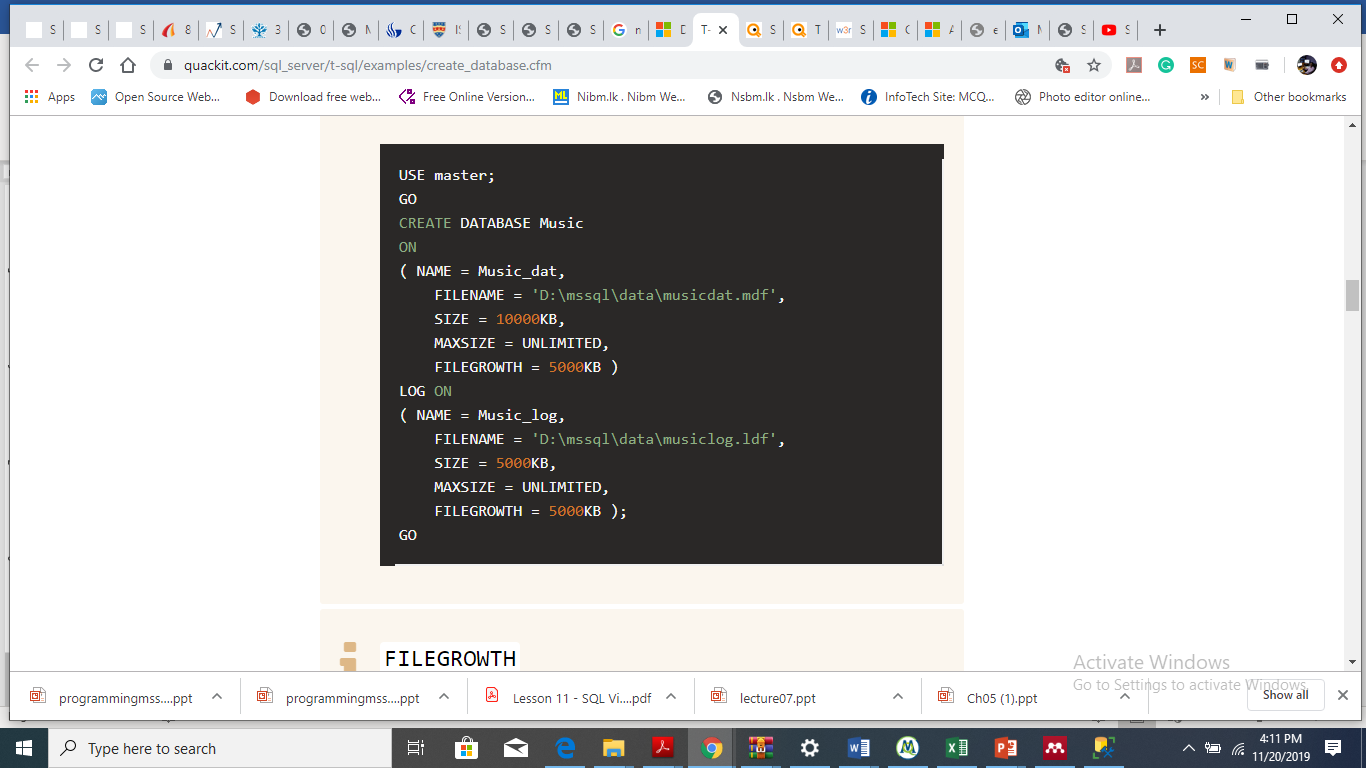
**logical\_file\_name:** The logical\_file\_name is the name used to refer to the physical file in all Transact-SQL statements. The logical file name must comply with the rules for SQL Server identifiers and must be unique among logical file names in the database. This is set by the NAME argument in ALTER DATABASE.

**Examples of creating data file and log file**

When you do this, NAME provides the logical name of the file, and FILENAME provides the physical path to the file on the operating system's file system.



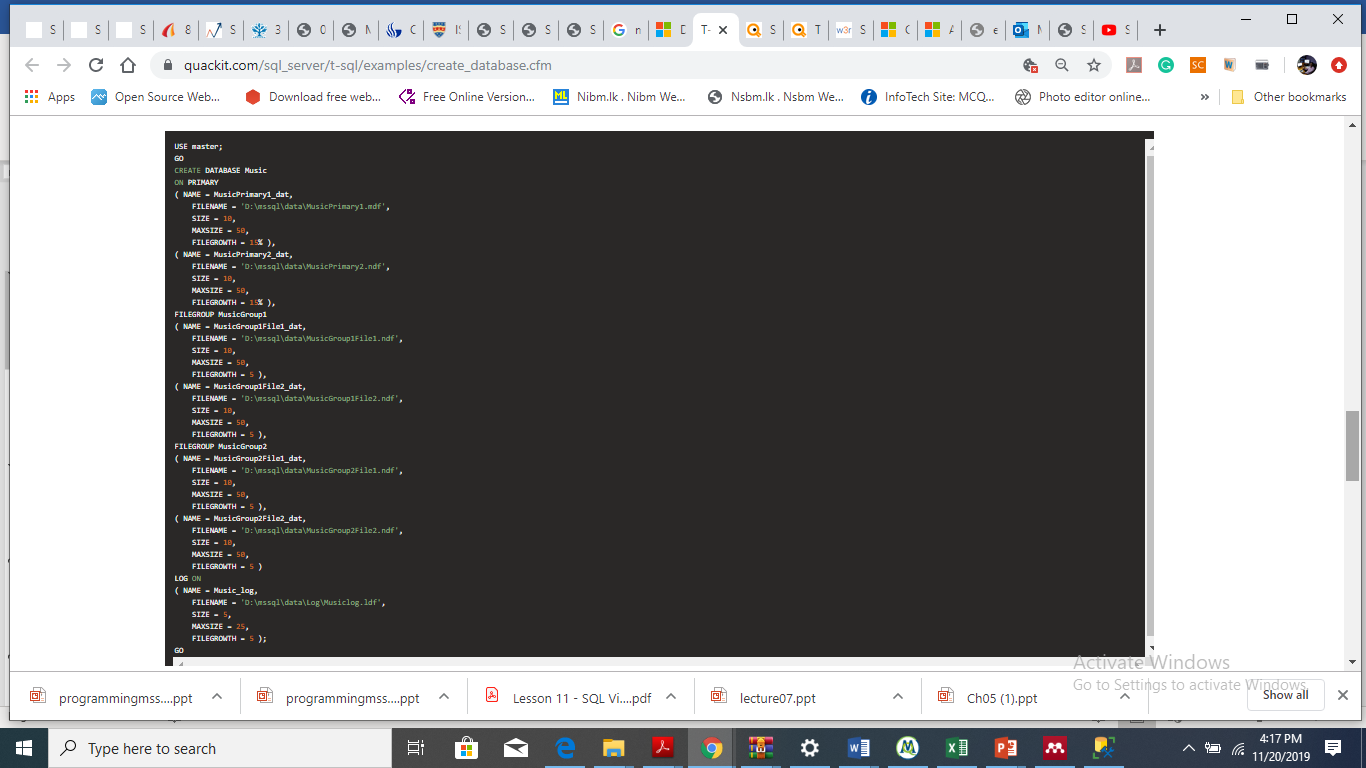
**Example with unlimited Maxfilesize**



## **Example for Multiple Data and Transaction Log Files**

## 

**Example for File groups**



## **More Examples**

### **A. Adding a file to a database**

The following example adds a 5-MB data file to the AdventureWorks2012 database.

USE master;

GO

ALTER DATABASE AdventureWorks2012

ADD FILE

(

NAME = Test1dat2,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\t1dat2.ndf',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

);

GO

### **B. Adding a filegroup with two files to a database**

The following example creates the filegroup Test1FG1 in the AdventureWorks2012 database and adds two 5-MB files to the filegroup.

USE master

GO

ALTER DATABASE AdventureWorks2012

ADD FILEGROUP Test1FG1;

GO

ALTER DATABASE AdventureWorks2012

ADD FILE

(

NAME = test1dat3,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\t1dat3.ndf',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

),

(

NAME = test1dat4,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\t1dat4.ndf',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

)

TO FILEGROUP Test1FG1;

GO

### **C. Adding two log files to a database**

The following example adds two 5-MB log files to the AdventureWorks2012 database.

USE master;

GO

ALTER DATABASE AdventureWorks2012

ADD LOG FILE

(

NAME = test1log2,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL13.MSSQLSERVER\MSSQL\DATA\test2log.ldf',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

),

(

NAME = test1log3,

FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL10\_50.MSSQLSERVER\MSSQL\DATA\test3log.ldf',

SIZE = 5MB,

MAXSIZE = 100MB,

FILEGROWTH = 5MB

);

GO

### **D. Removing a file from a database**

The following example removes one of the files added in example B.

USE master;

GO

ALTER DATABASE AdventureWorks2012

REMOVE FILE test1dat4;

GO

### **E. Modifying a file**

The following example increases the size of one of the files added in example B. The ALTER DATABASE with MODIFY FILE command can only make a file size bigger, so if you need to make the file size smaller you need to use DBCC SHRINKFILE.

USE master;

GO

ALTER DATABASE AdventureWorks2012

MODIFY FILE

(NAME = test1dat3,

SIZE = 200MB);

GO

This example shrinks the size of a data file to 100 MB, and then specifies the size at that amount.

USE AdventureWorks2012;

GO

DBCC SHRINKFILE (AdventureWorks2012\_data, 100);

GO

USE master;

GO

ALTER DATABASE AdventureWorks2012

MODIFY FILE

(NAME = test1dat3,

SIZE = 200MB);

GO

### **F. Moving a file to a new location**

The following example moves the Test1dat2 file created in example A to a new directory.

USE master;

GO

ALTER DATABASE AdventureWorks2012

MODIFY FILE

(

NAME = Test1dat2,

FILENAME = N'c:\t1dat2.ndf'

);

GO

### **G. Moving tempdb to a new location**

The following example moves tempdb from its current location on the disk to another disk location. Because tempdb is re-created each time the MSSQLSERVER service is started, you do not have to physically move the data and log files. The files are created when the service is restarted in step 3. Until the service is restarted, tempdb continues to function in its existing location.

1. Determine the logical file names of the tempdb database and their current location on disk.

SELECT name, physical\_name

FROM sys.master\_files

WHERE database\_id = DB\_ID('tempdb');

GO

1. Change the location of each file by using ALTER DATABASE.

USE master;

GO

ALTER DATABASE tempdb

MODIFY FILE (NAME = tempdev, FILENAME = 'E:\SQLData\tempdb.mdf');

GO

ALTER DATABASE tempdb

MODIFY FILE (NAME = templog, FILENAME = 'E:\SQLData\templog.ldf');

GO

1. Stop and restart the instance of SQL Server.
2. Verify the file change.

SELECT name, physical\_name

FROM sys.master\_files

WHERE database\_id = DB\_ID('tempdb');

1. Delete the tempdb.mdf and templog.ldf files from their original location.

### **H. Making a filegroup the default**

The following example makes the Test1FG1 filegroup created in example B the default filegroup. Then, the default filegroup is reset to the PRIMARY filegroup. Note that PRIMARY must be delimited by brackets or quotation marks.

USE master;

GO

ALTER DATABASE AdventureWorks2012

MODIFY FILEGROUP Test1FG1 DEFAULT;

GO

ALTER DATABASE AdventureWorks2012

MODIFY FILEGROUP [PRIMARY] DEFAULT;

GO

### **I. Adding a Filegroup Using ALTER DATABASE**

The following example adds a FILEGROUP that contains the FILESTREAM clause to the FileStreamPhotoDB database.

--Create and add a FILEGROUP that CONTAINS the FILESTREAM clause.

ALTER DATABASE FileStreamPhotoDB

ADD FILEGROUP TodaysPhotoShoot

CONTAINS FILESTREAM;

GO

--Add a file for storing database photos to FILEGROUP

ALTER DATABASE FileStreamPhotoDB

ADD FILE

(

NAME= 'PhotoShoot1',

FILENAME = 'C:\Users\Administrator\Pictures\TodaysPhotoShoot.ndf'

)

TO FILEGROUP TodaysPhotoShoot;

GO

The following example adds a FILEGROUP that contains the MEMORY\_OPTIMIZED\_DATA clause to the xtp\_db database. The filegroup stores memory optimized data.

--Create and add a FILEGROUP that CONTAINS the MEMORY\_OPTIMIZED\_DATA clause.

ALTER DATABASE xtp\_db

ADD FILEGROUP xtp\_fg

CONTAINS MEMORY\_OPTIMIZED\_DATA;

GO

--Add a file for storing memory optimized data to FILEGROUP

ALTER DATABASE xtp\_db

ADD FILE

(

NAME='xtp\_mod',

FILENAME='d:\data\xtp\_mod'

)

TO FILEGROUP xtp\_fg;

GO

**os\_file\_name:** The os\_file\_name is the name of the physical file including the directory path. It must follow the rules for the operating system file names. This is set by the FILENAME argument in ALTER DATABASE. For more information