`

Knowledge Base Article

Product Group: Software
Product: CMSW7400 - @ptitude Analyst / Microsoft SQL Server
Version: N/A

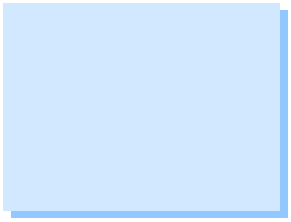
Abstract

This article lists some suggested tuning procedures for optimum performance in SKF @ptitude Analyst when using Microsoft SQL Server.

Overview

The tasks listed below can be done in order to improve database performance. This article was written for Microsoft SQL Server databases.

IMPORTANT NOTE: Create a backup of the database before attempting any of the following procedures:



`

Rebuild Indexes

Execute Rebuild Indexes on regular basis (once or twice a week) to improve database performance. This can also be scheduled for standard databases.

1. Log in to SQL Server Management Studio as skfuser1 (password cm).
2. Click on New Query.
3. Type in the following command:

```
Execute Rebuild_Index_All
```

4. Click on Execute.
5. The rebuild procedure will run. [Figure 1.1]

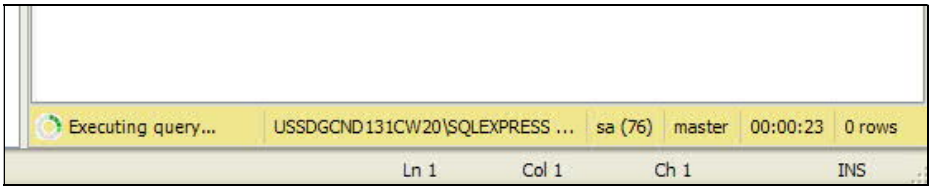


Figure 1.1. Executing query…

6. Once the procedure is complete, the output will be displayed. The message "Query executed successfully" will be shown, along with the time it took to complete the rebuild. [Figure 1.2] Depending on the size of the database, this process could take up to several minutes.

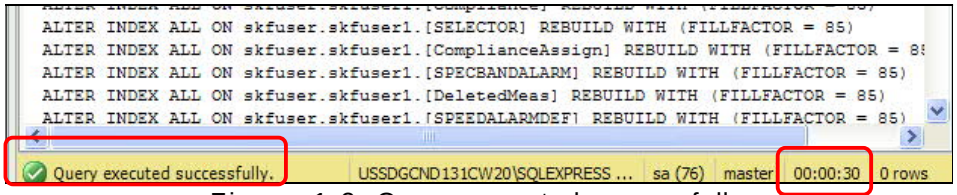


Figure 1.2. Query executed successfully

7. If the error message in Figure 1.3 is encountered, it means the procedure was not created. This could point to other problems.

`

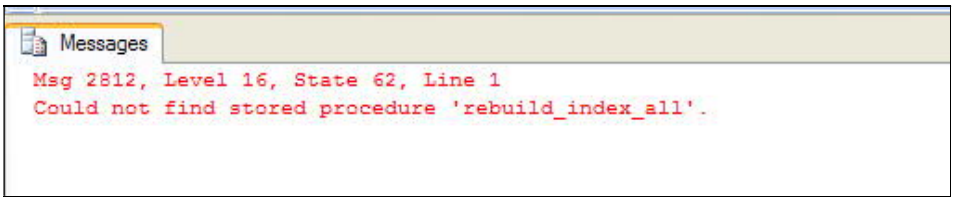


Figure 1.3. Example of an error message

`

Shrink Database and Files

This procedure will reduce space fragmentation and release space. The database cannot be made smaller than the minimum size of the database. The minimum size is the size specified when the database was originally created. For example, if a database was originally created with a size of 10 MB and grew to 100 MB, the smallest size the database could be reduced to is 10 MB, even if all the data in the database has been deleted.

The shrink operation cannot be executed on a database while the database is being backed up. Conversely, a database cannot be backed up while a shrink operation on the database is in process.

A shrink operation is most effective after an operation that creates lots of unused space, such as a truncate table or a drop table operation.

A shrink operation does not preserve the fragmentation state of indexes in the database, and generally increases fragmentation to a degree. This is another reason not to repeatedly shrink the database.

To view the current amount of free (unallocated) space in the database, a report can be run in Studio Manager to see the utilization of disk space within the database:

1. In Studio Manager, right-click on the skfuser database, then select Reports > Standard Reports > Disk Usage. [Figure 2.1]

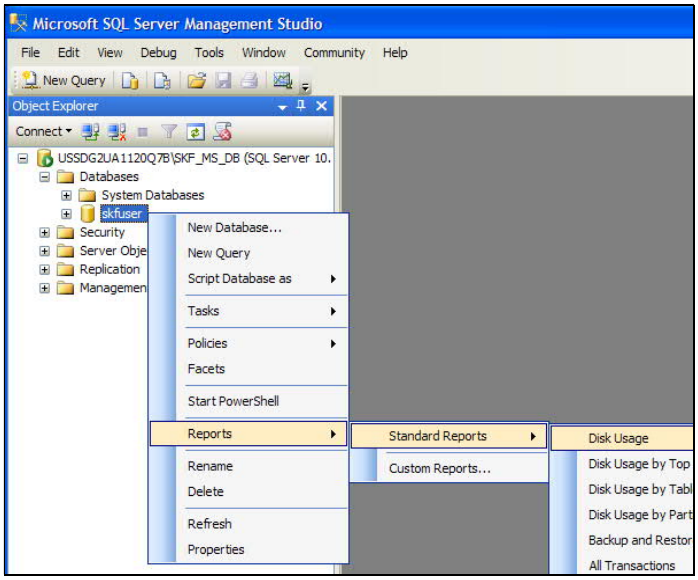


Figure 2.1. Disk Usage report

2. It will take a few minutes to run the report. A sample of the output is shown in Figure 2.2.

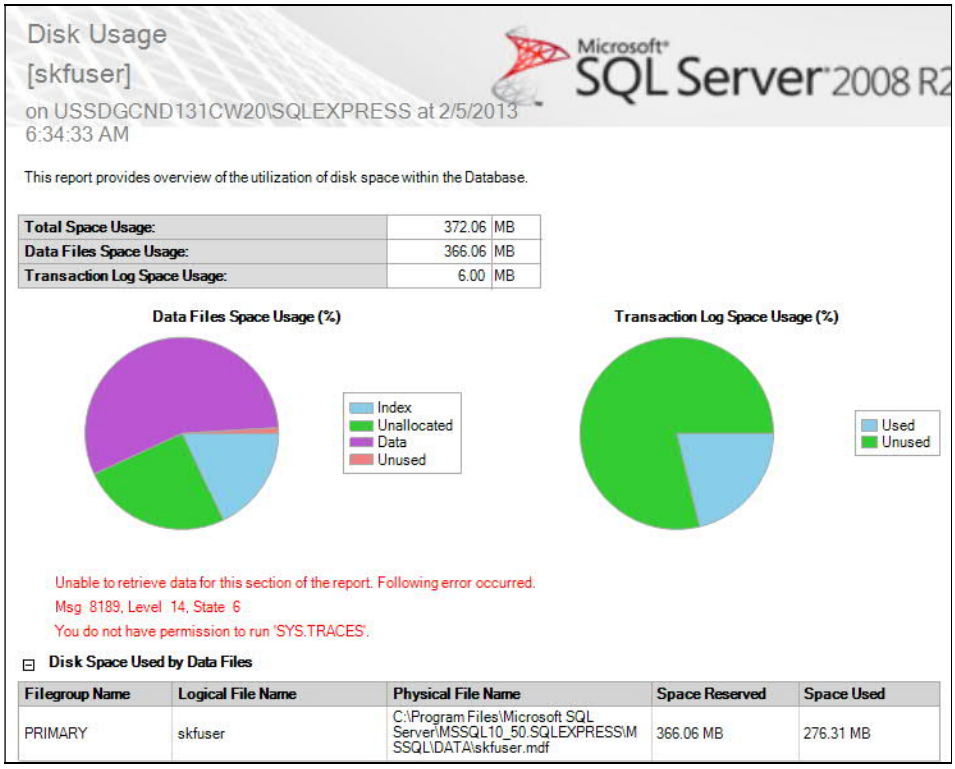


Figure 2.2. Sample of Disk Usage report output

Steps to Shrink a Database:

1. Create a backup of the database before performing this operation.

2. Close all connections to the database.

3. Log in to SQL Server Management Studio as sa user (default password is skf).

4. Right-click on the skfuser database, then select Tasks > Shrink > Database. [Figure 2.3]

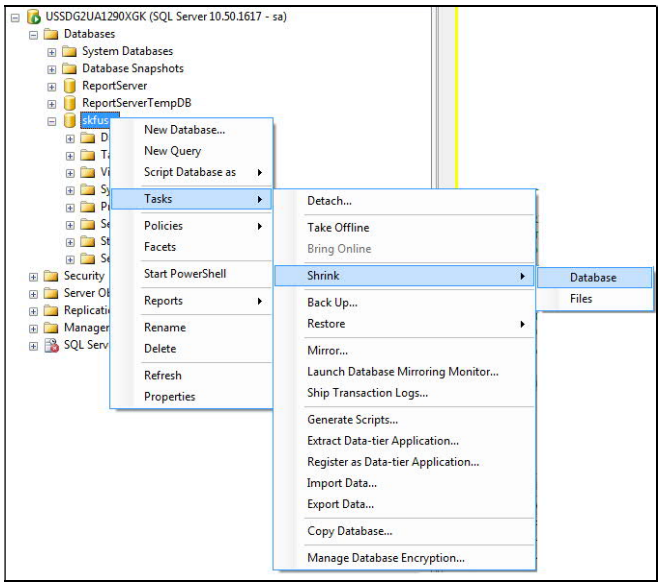


Figure 2.3. Tasks > Shrink > Database

5. The Shrink Database dialog will be displayed. [Figure 2.4] Click on OK to continue.

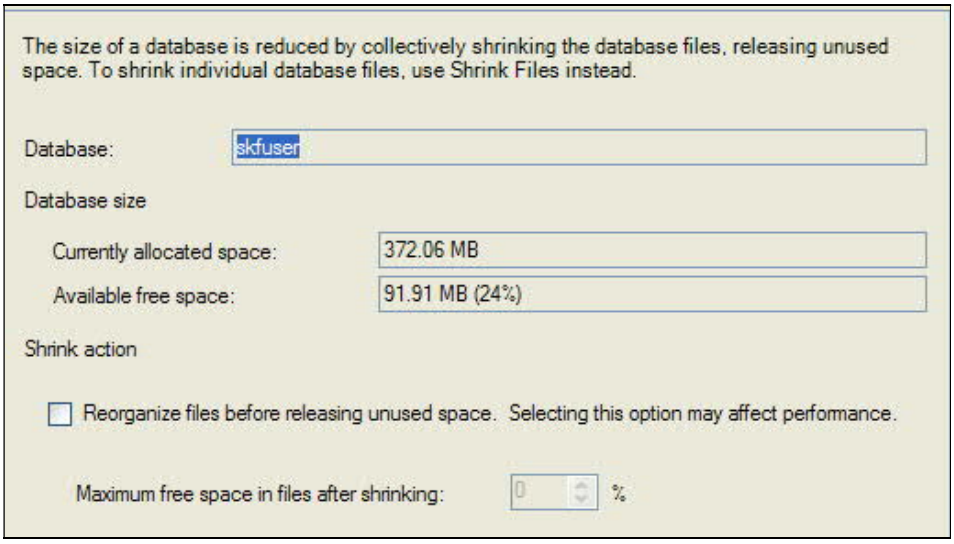


Figure 2.4. Shrink Database dialog

- Database: Displays the name of the selected database.
- Currently allocated space: Displays the total used and unused space for the selected database.
- Available free space: Displays the sum of free space in the log and data files of the selected database.

- Reorganize files before releasing unused space : By default, this option is not selected when the dialog is opened. If this option is selected, the user must specify a target percent option.

- Maximum free space in files after shrinking : Enter the maximum percentage of free space to be left in the database files after the database has been shrunk. Permissible values are between 0 and 99.

6. The screen will automatically close without indication if the shrink function was successful. Expect the .mdf file to be smaller if it was indeed fragmented.

7. After this task is complete, shrink the files by selecting Tasks > Shrink > Files (refer to Step 4).

8. The Shrink function may take a very long time (hours) for large files.

Data that is moved to shrink a file can be scattered to any available location in the file. This causes index fragmentation and can slow the performance of queries that search a range of the index. To eliminate the fragmentation, rebuild the indexes on the file after shrinking.

`

Check Log File Size

If there are a lot of uncommitted transactions taking place, the skfuser.ldf log file will become very large (larger than the skfuser.mdf file).

In one case, a customer had a skfuser.ldf log file that was 261 GB, when their skfuser.mdf file was only 25 GB. In the case of this customer, they were using IMx devices which took data every five minutes, but only wrote to the database once per hour.

The quickest way to fix this issue is to simply attach the database without the skfuser.ldf file:

1. Ensure all users have exited the program.
2. Stop all services (IMx Service, Transaction Server, Microlog Service).
3. Create a backup of the database.
4. Right-click on the skfuser database, then select Tasks > Detach... [Figure 3.1]

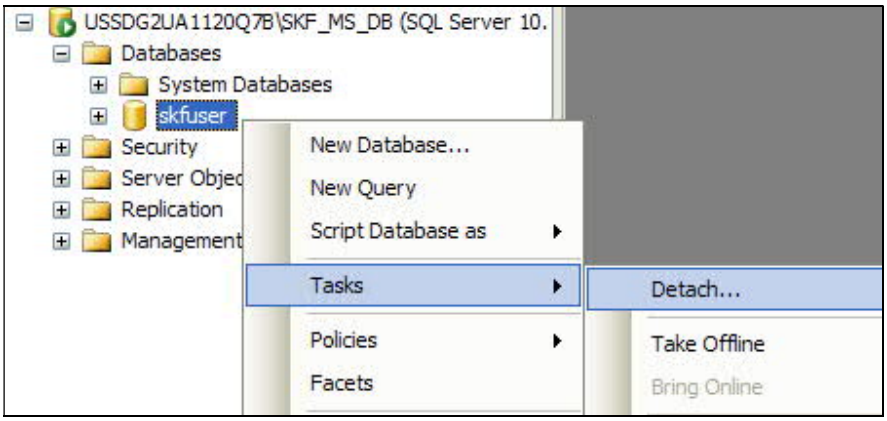


Figure 3.1. Detach database

5. Using Windows Explore, move or copy off the log file. [Figure 3.2]

Figure 3.2. Log file

6. The skfuser.mdf file will be left in the data directory. [Figure 3.3]

Figure 3.3. skfuser.mdf file remains

7. Next, attach the skfuser.mdf file in Studio Manager. [Figure 3.4]

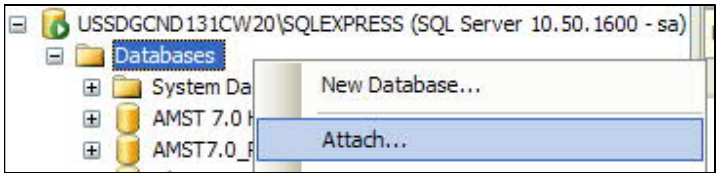


Figure 3.4. Attach database

8. Select the skfuser.mdf data file [Figure 3.5], then click OK.

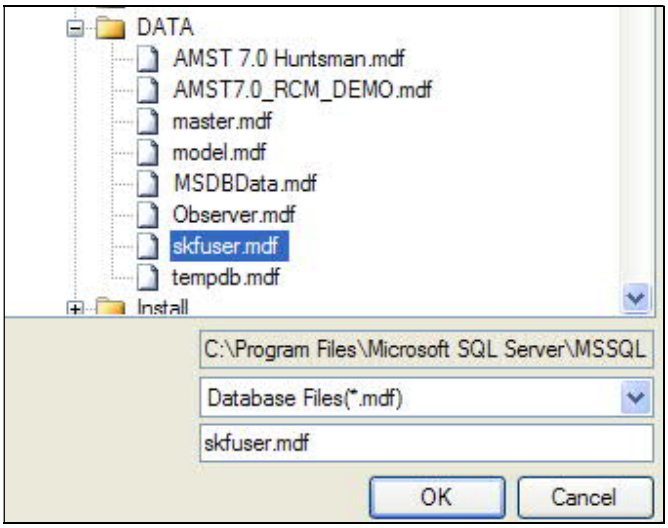


Figure 3.5. Locate skfuser.mdf file

9. Both files will be listed in the details window. The log file will need to be removed. To remove the log file, select skfuser.ldf [Figure 3.6] and click on Remove.

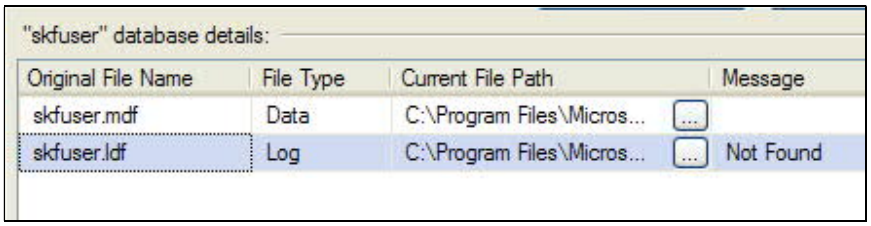


Figure 3.6. Remove skfuser.ldf log file

10. The skfuser.mdf will be the only data file remaining. [Figure 3.7] Click on OK.

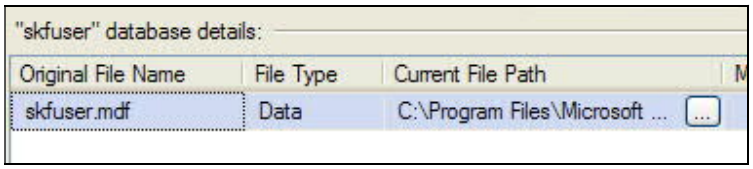


Figure 3.7. skfuser.mdf data file

11. A log file with the default name databasename_log.ldf will be created. [Figure 3.8]

Figure 3.8. Log file with default name

12. Finally, run the Rebuild_loginname_username_links.sql script. This script can be found in the SKF @ptitude Analyst installation folder. The default location is: C:\Program Files (x86)\SKF-RS\SKF @ptitude Analyst\DB Assist\SQL Server

`

Clean Up UTC Tables

This procedure will rebuild all UTC Tables from usertbl table. Usually this procedure gets created after dbupdate, but because it is only available on 2012 updates, it may have to be done manually. Disconnect ALL USERS first before starting.

For MR1 and later (built-in function, using dbTools file):

1. Log in to SQL Server Management Studio as skfuser1 (password cm).
2. Click on New Query.
3. Go to the SQL Server folder under SKF-RS\...\DB Assist and open the file “dbTools.sql” with Notepad. Copy the content to the new window in SQL Server Management Studio.
4. Click on Execute.
3. Remove the content then type in the following command:

```
execute dbTools_Rebuild_UTC_Tables
```

4. Click on Execute.

For earlier releases where this built-in functionality is not available:

1. Log in to SQL Server Management Studio as skfuser1 (password cm).
2. Click on New Query.
3. Type in the following command:

```
execute clear_utc_tables
```

4. Click on Execute.
5. The rebuild procedure will run and display a confirmation. [Figure 4.1]

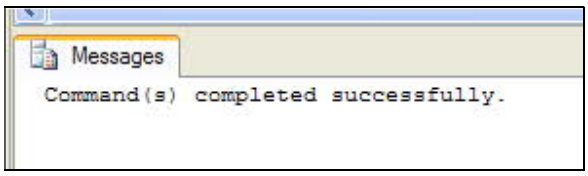


Figure 4.1. Command(s) completed successfully

`

Clean Up DeletedMeas Table (if XML import/export is not used)

DeletedMeas table is used for those customers who use XML files to export data to let other systems know that the attached measurements have been deleted and are not needed in the import.

1. Log in to SQL Server Management Studio as skfuser1 (password cm).

2. Click on New Query.

3. Type in the following command:

```
truncate table skfuser.skfuser1.DeletedMeas;
```

4. Click on Execute.

`

Check/Clean Up RouteWkspaceParents Table (pre MR1)

Check the table to see if it has duplicate ParentIds and how large the table is. It could contain thousands of duplicate records. The MR1 release will take care of managing this table in the future.

1. Log in to SQL Server Management Studio as skfuser1 (password cm).

2. Click on New Query.

3. Type in the following command:

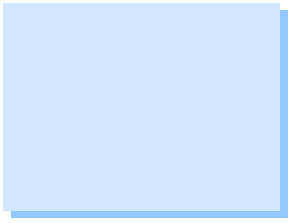
```
Select ParentId, count(*) DuplicateCount
From RouteWkspaceParents
Group by ParentId

Cleanup table:

Select distinct ParentId into RouteWkspaceParentsTemp
from RouteWkspaceParents

go
Truncate table RouteWkspaceParents
go
insert into RouteWkspaceParents
Select parentId from RouteWkspaceParentsTemp
go
Drop table RouteWkspaceParentsTemp
```

4. Click on Execute.

`

Periodically Restart SQL Service

If overall server performance is slow, try restarting SQL Server service. Restarting the service will clear and release memory and page caches that can affect the SQL server and database performance. There is no recommend time frame to restart the service.



`

Ensure Transaction Service is cleaning up records deleted from @ptitude Analyst

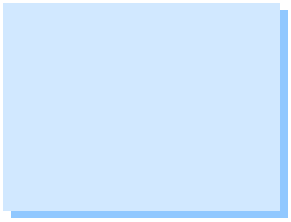
This operation is handled by the Transaction Server service, however, there are times when the Transaction Server can not catch up and must be helped.

This command will give the record count:

```
Select * from skfuser.skfuser1.TREEELEM where
PARENTID = 2147000000
```

This command will delete the records that are tagged for deletion:

```
Delete from skfuser.skfuser1.TREEELEM where
PARENTID = 2147000000
```



`

Check for Temp Tables

This should not be created, however, will be visible if @ptitude Analyst crashed when creating Workspaces (in most cases).

The tables will look like the example in Figure 5.1, named skfuser1.tempXXXXXX. The X's will be a date and time stamp when the table was created.

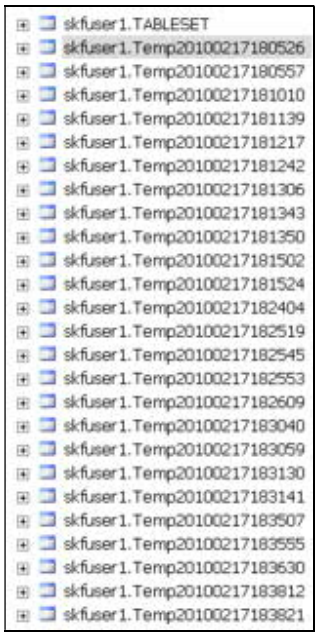


Figure 5.1. Temp tables

5. These temp tables can be deleted using Studio Manager. Right-click over a table name and select Delete. [Figure 5.2]

Figure 5.2. Delete temp tables

6. Repeat this process on the other temp tables that are present.

`

Check EventLog Table Size

This table holds all of the events that occur in @ptitude Analyst (logins, logouts, downloads, uploads, number of POINTs processed), therefore this table can grow to millions of records.

The Purging of the Event Log is handled by the SKF Monitor application. [Figure 6.1]

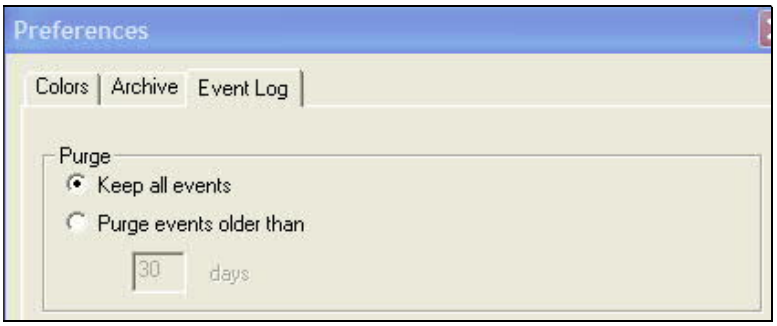


Figure 6.1. Preferences dialog

The options are:

- Keep all events: All events are saved and the table will continue to grow.
- Purge events older than number of days: All events older than the number of days entered will be purged. The Monitor must be running in order for the events to purge.

The EventLog table can be cleaned up using the following SQL commands:

```
delete from skfuser.skfuser1.EVENTLOG where
EVENTDTG like '2011%'
```

(The above command will delete all events from 2011.)

```
delete from skfuser.skfuser1.EVENTLOG where
EVENTDTG like '20111%'
```

(The above command will delete all events from January 2011.)

`

Other Items to Consider…

- Recommend the SQL Server be on a dedicated, high-performance machine.
    - Lots of RAM to help SQL cache as much data as possible
    - Fast hard drives
    - Avoid too machine virtual machines on the same box
    - Use multiple local drivers to breakup database files on separate files
    - Utilize RAID technology (RAID 5) recommended for I/O improvement
- Check RAM usage and availability. The more RAM, the better SQL Server functions. The recommendation for @ptitude Analyst is to dedicate at least 6 to 8 GB.
- Use SQL Server Standard or greater rather than SQL Express.
    - SQL Server express is limited to 1 GB of RAM, 1 Processor, and 10GB of hard drive space
- Take advantage of the Archive feature within @ptitude Analyst to prevent the database from growing out of control.
- Remember to periodically restart the SQL Server service (once every few months or as needed).
- Set up preferences in @ptitude Analyst to view current data rather than all measurements for normal data viewing.
- Recommend cleaning up routes and workspaces that are not used; this is very important.
- Use dbTools to defragment indexes.
    - Open SQL Server Management Studio and connect as SA user.
    - Open New Query window and type:

      execute Rebuild_SQL_Instance_Database_Indexes
    - Click Execute
    - This will rebuild all indexes on all databases within the instance at a default of 50% fragmentation or higher.
- Maintain EventLog table and make sure it does not get into tens of thousands of records.

- Clean up temp tables left on skfuser1 schema due to a database crash or error.
- Place database files on separate drives to improve I/O performance for system and Analyst databases.
    - Place skfuser1.mdf on local drive
    - Place skfuser1.ldf on another local drive.
    - Place tempdb yet on a third drive.
    - How to move a database file to a different drive.
    - Open SQL Server Management Studio and connect as SA user.
    - Stop all Analyst services.
    - Detach database.
    - Move the desired file to the designated local drive.
    - Use SQL Server Management Studio to attach.
    - Navigate to the skfuser.mdf file.
    - A message will be displayed [Figure 7] indicating file “Not Found.”

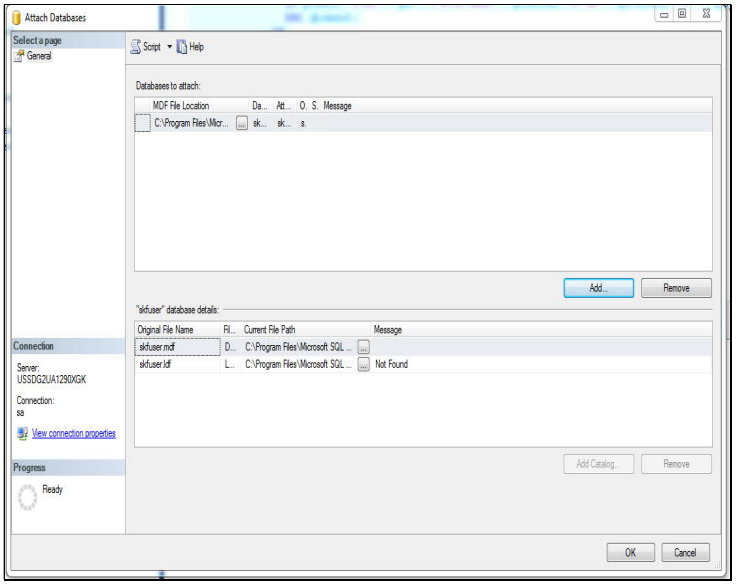


Figure 7. Not Found message

`

- Click on the "…" button next to it on the left and navigate to locate the file.

- Select the file and click the OK button.

Note:
If a privilege denied message is displayed, try repeating the attach process using Windows Authentication as in example in Figure 8 instead of SA user.

Figure 8. Logging in using Windows Authentication

It is possible SQL Server services (thus SA user also) lack write privileges to other local drives.

Windows Authentication will have full privileges to all local drives, assuming this process is executed by a DBA who is also an administrator on the server.

For further assistance, please contact the Technical Support Group by phone at 1-800-523-7514 option 8, or by e-mail at TSG-CMC@skf.com.