Upgrade Procedure

SpeechMiner draft

12/24/2014

Table of Contents

Welcome

The SpeechMiner Upgrade Guide provides the instructions required to upgrade SpeechMiner. These pages are valid for all SpeechMiner releases.

Upgrade Paths

SpeechMiner 8.0 to 8.1 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.0 to version 8.1.

Pre-upgrade Requirements

- Request the most recent release of the 8.5 software from your Genesys representative.
- Request the SpeechMiner license from Genesys Licensing.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Storage Requirements	To successfully complete the upgrade process, the data partition in the SQL server must have available disk space. The minimum required storage for the upgrade should be twice the size of the production database .mdf file.
Perform Pre-upgrade Tests	Since upgrading SpeechMiner can take several hours it is important to avoid delays. For this reason, it is recommended to test SMUpgrade before you begin the upgrade procedure. To test SMUpgrade, perform one of the following: • Create a copy of the database and send it to your Genesys counterpart. Genesys will test the SMUpgrade process in a lab environment. • Create a copy of the database on a separate SQL server. Provide your Genesys counterpart with the details of the SQL server and Genesys will test the SMUpgarde process on this temporary server.
Check for Customization	If any customizations were implemented on your DB, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.

Purging Old Data	Most systems have a data retention policy in place. Data (for example, audio, exploration data, and so on) that is older than the specified period of time is automatically deleted. In these cases, it is recommended to purge the old data before performing the upgrade. Deleting the data before the upgrade will reduce both the time it takes to run the upgrade process and the storage-space requirements.

Rollback Plan

To ensure that you can revert back to SpeechMiner 8.0, keep the 8.0 DB active on the server, rather than just keeping a backup file.

Do not delete the 8.0 data folders (`index`, `grammars`, etc.). Instead, configure the 8.1 system with new data folders. Creating new folders ensures that you will not loose 8.0 data.

Since the 8.0 DB and data folders are saved and available, back-out steps are not required if problems arise with the upgrade process before you uninstall 8.0. The 8.0 system should still be configured and functional.

After you uninstall SpeechMiner 8.0 and install SpeechMiner 8.1, the only way to revert back to 8.0 is to install 8.0 again and update the config files using SMConfig. However, since the DB and data folders would not have been deleted, they should be available and ready to use without changing the system configuration.

Upgrade Procedure

The following table lists the approximate times required to complete the upgrade steps:

Step	Time
Stop the system (step 1)	15 minutes
Backup the database (step 2)	120 minutes
Create target database (step 3)	30 minutes
Run SMUpgrade (steps 6 to 13)	10 to 20 hours
Configuring and starting the system (steps 14 to end)	60 minutes

1. Using `SMConfig->Services->Stop Services`, stop the 8.0 system.

2. Create a copy of the source DB and upgrade it to the latest build:
   The source DB must be version 8.0. Refer to the `versionTb1` table to determine the correct version.
   If the source DB is not the latest build and you do not want to update it to the latest build, create a copy of the source DB and update the copy to the latest build.

   > **Important**
   >
   > These steps are necessary because the 8.0 DB schema needs to be updated to the latest schema in order for the rest of the upgrade process to succeed.

   Use the copy of the source DB as the baseline for the 8.1 upgrade.

   a. Back up the 8.0 index folder to a backup folder (see Configuring SpeechMiner—Index).
   b. Create a copy of the source DB (back up the DB and then restore it in another location).
   c. Configure the copy of the DB with the Index backup. This will be the baseline for the upgrade.
   d. Implement the schema changes on the baseline DB to bring the schema into line with the latest 8.0 build version. For this step you will need assistance from Genesys Customer Care. Contact Customer Care for assistance.
3. Create the 8.1 target DB as follows:
   - Manually—Refer to **Installing the SpeechMiner Database > Manual** tab.

   Or
   - Setup Wizard— Refer to **Installing the SpeechMiner Database > Setup Wizard** tab.
4. If the MS-SQL server is an Enterprise Edition, run `EXEC sp_create_DB_storage_partitions` on the target database.
5. If your source and target databases are on different servers, make sure the servers are linked in both directions, using the stored procedures `sp_addlinkedserver` and `sp_addlinkedsrvlogin`, as needed.
6. Install and run SMUpgrade (to migrate the data from the 8.0 DB to the 8.1 DB), as follows:

Prerequisites:

- When migrating a large database, make sure that the hard drive that hosts the target database has enough storage space.

Usage

a. Query the `versionTbl` table to ensure that your 8.0 source database is updated to the latest 8.0 schema.
b. Verify that your recovery model is either `Simple` or `Bulk-logged`. To determine which recovery model you have, right click `db > properties > options > recovery model`.
c. Use the SpeechMiner Installer to install the `SMUpgrade` component. It is recommended to install and run the SMUpgrade component on the SQL server.
d. Configure the following in the `\utopy\tools\bin\release\SMUpgrade.exe.config` file:
    - file locations
    - tables to skip (comma separated list)
    - number of threads running concurrently on a large table
    - bulk copy usage

    It is recommended that the `bulk-load` folder be located on the SQL server. The `bulk-load` folder must be shared and the user running the SQL server service must have full control over it.
    If you do not set the bulk copy usage, the `callAudioTbl` upgrade will take up to two times longer on a large DB.
    Only use the `skip-tables` configuration if specifically requested by Genesys Customer Care

```
 <appSettings>
<add key="ErrorLogFile"
value=".\SMUpgradeLog.txt" />
<add key="LogFile"
```

```
value=".\tableLog.txt" >
<add key="TimingsFile"
value=".\tableTimings.txt" />
<add key="SkipTables"
value="callTasksTbl" />
<add key="NumThreads" value="10" />

</appSettings>
```

e. Run `SMUpgrade.exe`.
Log in and select the appropriate 8.0 source and 8.1 destination databases.

> **Important**
>
> It is highly recommended to use the `sa` credentials or a user account with bulk insert permissions.
>
> The user account must belong to the `db_owner` role in the target database. By default, the `DBUser` does not include the `db_owner` role.

f. The GUI shades the tables as follows:
   - Green—The table is finished. Both the source and the target DBs contain the same number of records. This conclusively shows that either the data has been copied or there is no data to copy.
   - Yellow—The number of records in the source and target DBs is not indicative of whether the

data in both DBs is identical or not. It is therefore not known whether the table is finished or not.
- Red—The table is not finished. The number of records in the source and target databases are not the same, and indicating that the data has not been copied in full.

g. Click `Full Upgrade` to run the upgrade, or `Resume Last` if your previous upgrade was interrupted.

   Resuming the last upgrade will shorten the time to run, but might cause problems. To avoid such problems, restore the database again and run the full upgrade.
   You can also define tables to skip in the configuration file. Every step in the upgrade process is shown in the GUI.
   You can stop the upgrade by clicking the `Close` button. You will be prompted to confirm your action. Note that the window closes immediately, but the process still runs for a while, as it needs to re-enable the indexes it disables when it starts running.
   The time each step took is written to the `TimingsFile`. The location of this file is defined in the configuration file.
   IN CASE OF FAILURE: Review all status, error and excepetion notifications in the `ErrorLogFile`.

h. Continue with the upgrade instructions below.

7. If the SpeechMiner Maintenance Job exists, and the `Update time table` step is included, delete the `Update time table` step. Make sure the last step in the job is set to quit the job upon both success and failure.
8. Optional: Uninstall 8.0 from all servers. The two versions (8.0 and 8.1) cannot be running side by side at the same time. Only one version can be registered as the active SpeechMiner service on each server. The installation binaries can be left on the server.
9. Install the 8.1 platform on all servers.
10. Install 8.1 Web on the Web server.
11. Install 8.1 SMART on users' desktops, as required.
12. Deploy SQLCLR on the DB server.

13. Update the package folders with the 8.1 `.gram` files. The `.gram` files are located in the `<Installation Folder>/Support/Grammars`.
    Alternatively, if you have not made changes to any file in these folders, you can delete their content completely. SMConfig will copy the grammar files to `<Installation folder>/Support/Grammars`.
14. Run `SMConfig`.
    a. Configure the **Sites & Machines**, panel as necessary, and save the changes. Make sure you save this panel even if you have not made any changes.
    b. Configure the **Services** panel and save the changes. Do not start any of the services.
    c. Configure the **Index** panel and save the changes.
    d. Update the SpeechMiner license with the new 8.1 licenses provided by Genesys Licensing.
    e. In the **Reports** panel, update the `MRSLibrary.dll` on the report server.
    f. Deploy the reports to the report server.
15. Start SMART and perform the following:
    a. Right-click on each active Program icon and choose `Activate program`. This will mark all the Programs, Topics, and Categories as changed.
    b. Click the `Apply` button.
    c. In the new `Apply` popup window, choose `Apply all`.
    d. Click the `Apply` button.
16. Using `SMConfig`, start the `UPlatform` services on all the servers.
17. Update the Stored Procedures by coping any existing custom Stored Procedures from the 8.0 DB to the 8.1 DB.
    It is not necessary to copy Stored Procedures that are used by gauges, and are in the `GaugeWidgetProcedures` table, because they are copied automatically.
18. Open the SpeechMiner web-based interface and test the functionality.
19. Update the Database Jobs:
    - All database jobs that point to the 8.0 DB should be changed to point to the new 8.1 DB. Examples of DB jobs that might need to be changed:
      - DB maintenance job
      - `sp_agentFilterCleanByDays`
      - `sp_updateUntilYesterdayMaxChannels`

    To change a DB job, it is recommended that you edit the Job Step property using the SQL Management studio.
20. In the SpeechMiner web-based interface, manually reschedule and 8.0 reports that should continue to run on a scheduled basis.

Deprecated Reports

The following reports were deprecated in 8.1:

- Audit Analysis v4
- Call List v4
- First Topic Distribution v2
- MINI_System Load
- Monitor System
- Predictive Elements v4
- Program Distribution v4
- Topic Call List v2
- User Management

SpeechMiner 8.1 to 8.5 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.1 to version 8.5.

Pre-upgrade Requirements

- Request the most recent release of the 8.5 software from your Genesys representative.
- Request the SpeechMiner license from Genesys Licensing.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Storage Requirements	To successfully complete the upgrade process, the data partition in the SQL server must have available disk space. The minimum required storage for the upgrade should be twice the size of the production database .mdf file.
Perform Pre-upgrade Tests	Since upgrading SpeechMiner can take several hours it is important to avoid delays. For this reason, it is recommended to test SMUpgrade before you begin the upgrade procedure. To test SMUpgrade, perform one of the following: • Create a copy of the database and send it to your Genesys counterpart. Genesys will test the SMUpgrade process in a lab environment. • Create a copy of the database on a separate SQL server. Provide Genesys with the details of the SQL server and Genesys will test the SMUpgarde process on this temporary server.
Check for Customization	If any customizations were implemented on your DB, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.
Purging Old Data	Most systems have a data retention policy in place. Data (for example, audio,exploration data, etc.) that is older than the specified period of time

is automatically deleted. If you do not have a data retention policy in place, it is recommended to determine what data should be saved and what can be discarded, since deleting the data before the upgrade will reduce both the time it takes to run the upgrade process and the storage-space requirements.

Rollback Plan

To ensure that you can revert back to SpeechMiner 8.1, keep the 8.1 DB active on the server, rather than just keeping a backup file.

Do not delete the 8.1 data folders (`index`, `grammars`, etc.). Instead, configure the 8.1 system with new data folders. Create the following new folders to ensure that you will not lose 8.1 data:

- Create the following empty folders:
  - Input
  - Interaction Receiver Input
  - Filtered
  - Index (this folder will be populated during the upgrade procedure).
- Copy the content of the following existing folders to new folders with the same name:
  - Store
  - Grammer
  - Backup

For detailed information about the folders you should create, refer to Required Folders

Since the 8.1 DB and data folders are saved and available, back-out steps are not required if problems arise with the upgrade process before you uninstall 8.1. The 8.1 system should still be configured and functional.

After you uninstall SpeechMiner 8.1 and install SpeechMiner 8.5, the only way to revert back to 8.1 is to install 8.1 again and update the config files using SMConfig. However, since the DB and data folders would not have been deleted, they should be available and ready to use without changing the system configuration.

Upgrade Procedure

The following table lists the approximate times required to complete the upgrade steps:

Step	Time

Stop the system (step 1)	15 minutes
Backup the database (step 2)	120 minutes
Create target database (step 3)	30 minutes
Run SMUpgrade (steps 6 to 13)	10 to 20 hours
Configuring and starting the system (steps 14 to end)	60 minutes

1. Using `SMConfig->Services->Stop Services`, stop the 8.1 system.
2. Create a copy of the source DB and upgrade it to the latest build:
   The source DB must be version 8.1. Refer to the `versionTbl` table to determine the correct version.
   If the source DB is not the latest build and you do not want to update it to the latest build, create a copy of the source DB and update the copy to the latest build.

   > **Important**
   >
   > These steps are necessary because the 8.1 DB schema needs to be updated to the latest schema in order for the rest of the upgrade process to succeed.

   Use the copy of the source DB as the baseline for the 8.5 upgrade.

   a. Back up the 8.1 index folder to a backup folder (see Configuring SpeechMiner—Index).
   b. Create a copy of the source DB (back up the DB and then restore it in another location).
   c. Configure the copy of the DB with the Index backup. This will be the baseline for the upgrade.
   d. Implement the schema changes on the baseline DB to bring the schema into line with the latest 8.1 build version. For this step you will need assistance from Customer Care.
3. Create the 8.5 target DB as follows:
   - Manually—Refer to **Installing the SpeechMiner Database > Manual** tab.

   Or
   - Setup Wizard— Refer to **Installing the SpeechMiner Database > Setup Wizard** tab.

4. If the MS-SQL server is an Enterprise Edition, run `EXEC sp_create_DB_storage_partitions` on the target database.
5. If your source and target databases are on different servers, make sure the servers are linked in both directions, using the stored procedures `sp_addlinkedserver` and `sp_addlinkedsrvlogin`, as needed.
6. Install and run SMUpgrade (to migrate the data from the 8.1 DB to the 8.5 DB), as follows:

Prerequisites:

- When migrating a large database, make sure that the hard drive that hosts the target database has enough storage space.

Usage

a. Query the `versionTbl` table to ensure that your 8.1 source database is updated to the latest 8.1 schema.
b. Verify that your recovery model is either `Simple` or `Bulk-logged`. To determine which recovery model you have, right click `db > properties > options > recovery model`.
c. Use the SpeechMiner Installer to install the `SMUpgrade` component. It is recommended to install and run the SMUpgrade component on the SQL server.
d. Configure the following in the `\utopy\tools\bin\release\SMUpgrade.exe.config` file:
    - file locations
    - tables to skip (comma separated list)
    - number of threads running concurrently on a large table
    - bulk copy usage

    It is recommended that the `bulk-load` folder be located on the SQL server. The `bulk-load` folder must be shared and the user running the SQL server service must have full control over it.
    If you do not set the bulk copy usage, the `callAudioTbl` upgrade will take up to two times longer on a large DB.

Only use the `skip-tables` configuration if specifically requested by Genesys Customer Care.

```
 <appSettings>
<add key="ErrorLogFile"
value=".\SMUpgradeLog.txt" />
<add key="LogFile"
value=".\tableLog.txt" >
<add key="TimingsFile"
value=".\tableTimings.txt" />
<add key="SkipTables"
value="callTasksTbl" />
<add key="NumThreads" value="10" />

</appSettings>
```

e. Run `SMUpgrade.exe`.
Log in and select the appropriate 8.1 source and 8.5 destination databases.

> **Important**
>
> It is highly recommended to use the `sa` credentials or a user account with bulk insert permissions.
>
> The user account must belong to the `db_owner` role in the target database. By default, the `DBUser` does not include the `db_owner` role.

f. The GUI shades the tables as follows:
   - Green—The table is finished. Both the source and the target DBs contain the same number of

records. This conclusively shows that either the data has been copied or there is no data to copy.

- Yellow—The number of records in the source and target DBs is not indicative of whether the data in both DBs is identical or not. It is therefore not known whether the table is finished or not.
- Red—The table is not finished. The number of records in the source and target databases are not the same, and indicating that the data has not been copied in full.

g. Click `Full Upgrade` to run the upgrade, or `Resume Last` if your previous upgrade was interrupted.

   Resuming the last upgrade will shorten the time to run, but might cause problems. To avoid such problems, restore the database again and run the full upgrade.
   You can also define tables to skip in the configuration file. Every step in the upgrade process is shown in the GUI.
   You can stop the upgrade by clicking the `Close` button. You will be prompted to confirm your action. Note that the window closes immediately, but the process still runs for a while, as it needs to re-enable the indexes it disables when it starts running.
   The time each step took is written to the `TimingsFile`. The location of this file is defined in the configuration file.
   IN CASE OF FAILURE: Review all status, error and exception notifications in the `ErrorLogFile`.

h. Continue with the upgrade instructions below.

7. If the SpeechMiner Maintenance Job exists, and the `Update time table` step is included, delete the `Update time table` step. Make sure the last step in the job is set to quit the job upon both success and failure.
8. Optional: Uninstall 8.1 from all servers. The two versions (8.1 and 8.5) cannot be running side by side at the same time. Only one version can be registered as the

active SpeechMiner service on each server. The installation binaries can be left on the server.

9. Install the 8.5 platform on all servers.
10. Install 8.5 Web on the Web server.
11. Install 8.5 SMART on users' desktops, as required.
12. Deploy SQLCLR on the DB server.
13. Update the package folders with the 8.5 `.gram` files. The `.gram` files are located in the `<Installation Folder>/Support/Grammars`.
    Alternatively, if you have not made changes to any file in these folders, you can delete their content completely. SMConfig will copy the grammer files to `<Installation folder>/Support/Grammars`.
14. Run `SMConfig`.
    a. Configure the **Sites & Machines**, panel as necessary, and save the changes. Make sure you save this panel even if you have not made any changes.
    b. Configure the **Services** panel and save the changes. Do not start any of the services.
    c. Configure the **Index** panel and save the changes.
    d. Update the SpeechMiner license with the new 8.5 licenses provided by Genesys Licensing.
    e. In the **Reports** panel, update the `MRSLibrary.dll` on the report server.
    f. Deploy the reports to the report server.
15. Start SMART and perform the following:
    a. Right-click on each active Program icon and choose `Activate program`. This will mark all the Programs, Topics, and Categories as changed.
    b. Click the `Apply` button.
    c. In the new `Apply` popup window, choose `Apply all`.
    d. Click the `Apply` button.
16. Using `SMConfig`, start the `UPlatform` services on all the servers.
17. Update the Stored Procedures by copying any existing custom Stored Procedures from the 8.1 DB to the 8.5 DB.
    It is not necessary to copy Stored Procedures that are used by gauges, and are in the `GaugeWidgetProcedures` table, because they are copied automatically.
18. Open the SpeechMiner web-based interface and test the functionality.
19. Update the Database Jobs:
    - All database jobs that point to the 8.1 DB should be changed to point to the new 8.5 DB. Examples of DB jobs that might need to be changed:
      - DB maintenance job
      - `sp_agentFilterCleanByDays`
      - `sp_updateUntilYesterdayMaxChannels`

    To change a DB job, we recommend that you edit the Job Step property using the SQL Management studio.

20. In the SpeechMiner web-based interface, manually reschedule 8.1 reports that should continue to run on a scheduled basis.

Deprecated Reports

The following reports were deprecated in 8.5:

- Audit Analysis v4
- Call List v4
- First Topic Distribution v2
- MINI_System Load
- Monitor System
- Predictive Elements v4
- Program Distribution v4
- Topic Call List v2
- User Management.

SpeechMiner 8.5.0 to 8.5.001 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.5.0 to version 8.5.001

Pre-upgrade Requirements

- Request the newest 8.5.001 software from Genesys.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Check for Customization	If any customizations were implemented on your database, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.

Time Requirements

The following table lists the approximate times required to complete the upgrade steps:

Step	Time
Stop the system (step 1)	10 minutes
Backup the database (step 3)	15 minutes (the larger the database the more time this step will take).
Configuring and starting the system (steps 13 to end)	40 minutes

Upgrade Procedure

1. Close all browsers and SpeechMiner applications.
2. Stop the **Uplatform** service.
3. Backup the SpeechMiner 8.5.0 database in the SQL server.
4. Uninstall SpeechMiner 8.5.0 (build 7055).

5. Copy the entire **FullInstaller** folder to your local server.
6. Install the new SpeechMiner 8.5.001 (build 7104).
7. Reboot your machine.
8. Perform SpeechMiner 8.5.0 database changes on the SQL Manager by running all the SpeechMiner 8.5.0 database commands in the SQL query window. To receive the relevant commands, see Database Changes - Commands.
9. Run the 8.5.001 database update script (not required for recording only installations). To receive the relevant script. To receive the script click 8.5.001 Database Script.
10. Deploy SQLCLR: Via SQL management run the commands that are in C:\Program Files (x86)\Genesys\Software\Support\sqlclr.sql in the SpeechMiner 8.5.0 database.
11. Run SMConfig and connect to the database that was upgraded.
12. Click **Save** in **Sites & Machines**.
13. Deploy the reports (not required for recording only installations).
14. In the **SpeechMiner Configuration Tool 8.5.0 > Services** select the following:
    - Under **Services** select:
      - **Create Performance Counters**
      - **Register services**
      - **Update config files**
      - **Encrypt config files**
    - Under **Select/Deselect All** select the relevant machine.
    - Select **Restart Services** and select **change status to run** from the drop down list.
15. Open the SpeechMiner Web and check its functionality.

8.5.0 to 8.5.001 Database Changes - Commands

Copy the following commands and run them in the SQL query window for the SpeechMiner 8.5.0 to 8.5.0.1 Upgrade Procedure:

```
ALTER TABLE dbo.segmentAudioTbl ADD arrivalTime int NULL
GO

GO
ALTER TABLE [dbo].[coachingStaticCallListCalls] DROP CONSTRAINT
[FK_coachingStaticCallListCalls_callMetaTbl]
GO

GO
```

```
GO

ALTER TABLE [dbo].[agentEntityTbl] DROP CONSTRAINT
[IX_agentEntityTbl]
GO


GO

GO

IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[sp_agentFilterCleanByDays]') AND type in (N'P',
N'PC'))
DROP PROCEDURE [dbo].[sp_agentFilterCleanByDays]
GO
ALTER PROCEDURE [dbo].[sp_rebuild_partition_help_tables]

WITH EXECUTE AS SELF

  AS



  BEGIN TRAN

          TRUNCATE TABLE partitionProgramTbl

        insert into partitionprogramTbl

        select distinct b.partitionid,a.programid

        from (select callId,programId from callMetaTbl union select
textId as callId,programId from TextData) a, callPartitionTbl as b

        where a.callid=b.callid


  COMMIT

IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[partitionAgentTbl]') AND type in (N'U'))
```

```
DROP TABLE [dbo].[partitionAgentTbl]
GO



GO
ALTER TABLE WordPrecision
ADD  "wer" float(20)
ALTER TABLE WordPrecision
ADD  "recall" float(20)
GO

GO

ALTER TABLE NGProbs1
ADD  "backoff" float(20)
ALTER TABLE NGProbs2
ADD  "backoff" float(20)
ALTER TABLE NGProbs3
ADD  "backoff" float(20)
ALTER TABLE NGProbs4
ADD  "backoff" float(20)
GO

GO
IF NOT EXISTS(select * from sys.columns where Name =
N'isBackupConfigurationServer' and Object_ID =
Object_ID(N'ComputerList'))
BEGIN
ALTER TABLE [dbo].[ComputerList] ADD isBackupConfigurationServer
bit NOT NULL DEFAULT 0
END


GO
CREATE TABLE wildcardGrammars
(
languageId int NOT NULL,
[key] nvarchar(50) NOT NULL,
value nvarchar(50) NOT NULL,
PRIMARY KEY (languageId,[key])
```

```
);

GO

INSERT INTO wildcardGrammars values
(0,'[alphanum]','alphanum'),
(0,'[boolean]','boolean'),
(0,'[ccexpdate]','ccexpdate'),
(0,'[creditcard]','creditcard'),
(0,'[currency]','currency'),
(0,'[date]','date'),
(0,'[digits]','digits'),
(0,'[number]','number'),
(0,'[phone]','phone'),
(0,'[socialsecurity]','socialsecurity'),
(0,'[time]','time'),
(0,'[voiceenroll]','voiceenroll'),
(0,'[zipcode]', 'zipcode'),
(2,'[alphanum]','alphanum'),
(2,'[boolean]','boolean'),
(2,'[ccexpdate]','ccexpdate'),
(2,'[creditcard]','creditcard'),
(2,'[currency]','currency'),
(2,'[date]','date'),
(2,'[digits]','digits'),
(2,'[number]','number'),
(2,'[phone]','phone'),
(2,'[time]','time'),
(2,'[voiceenroll]','voiceenroll'),
(2,'[zipcode]','zipcode');

GO

GO
ALTER TABLE NGProbs1
DROP COLUMN backoff
ALTER TABLE NGProbs2
DROP COLUMN backoff
ALTER TABLE NGProbs3
DROP COLUMN backoff
ALTER TABLE NGProbs4
DROP COLUMN backoff
```

```
GO

GO
ALTER TABLE RecognitionLanguages
ADD "WordConfidence" varchar(100)
GO

Update RecognitionLanguages
SET WordConfidence = 'wordconfidence'
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_German'  where index1 = 6
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_Catalan'  where index1 = 13
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_English'  where index1 in
(0,3,14)
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_Spanish'  where index1 in (1,2)
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_Portuguese'  where index1 = 11
UPDATE RecognitionLanguages
SET WordConfidence = 'wordconfidence_French'  where index1 = 5
GO

CREATE TABLE WordConfidenceInterpolation
(
Language int,
wer varchar(255),
precision varchar(255),
recall varchar(255)
)

grant SELECT on WordConfidenceInterpolation to Platform

INSERT INTO WordConfidenceInterpolation values(0,'-0.0356
0.8034','0.0431 0.3559','0.0272 0.5093')
INSERT INTO WordConfidenceInterpolation values(1,'-0.0383
0.7144','0.0226 0.5246','0.0338 0.3492')
INSERT INTO WordConfidenceInterpolation values(2,'-0.0369
0.8605','0.0418 0.3515','0.0343 0.1833')
INSERT INTO WordConfidenceInterpolation values(3,'-0.0328
0.7378','0.0395 0.3265','0.0246 0.3305')
```

```
INSERT INTO WordConfidenceInterpolation values(5,'-0.0334
0.9805','0.0386 0.1734','0.0386 0.1734')
INSERT INTO WordConfidenceInterpolation values(6,'-0.0179
0.6302','0.0295 0.3729','0.018 0.3883')
INSERT INTO WordConfidenceInterpolation values(11,'-0.0203
0.3161','0.0339 0.3161','0.0167 0.4534')
INSERT INTO WordConfidenceInterpolation values(13,'-0.0356
0.8034','0.0431 0.3559','0.0272 0.5093')
INSERT INTO WordConfidenceInterpolation values(14,'-0.033
0.6866','0.043 0.3835','0.0379 0.3249')
GO

GO
GRANT select ON [dbo].[wildcardGrammars] TO
[SMART],[SMARTCompileGrammar],[InteractionReceiver],[Platform],[SMConfig],[We

GO

GO

IF  EXISTS (SELECT * FROM sys.indexes WHERE object_id =
OBJECT_ID(N'[dbo].[callPartitionTbl]') AND name =
N'IX_callPartitionTbl_partitionId')
DROP INDEX [IX_callPartitionTbl_partitionId] ON
[dbo].[callPartitionTbl] WITH ( ONLINE = OFF )
GO



GO
Update RecognitionLanguages
set Display='South African English',
    NuanceRecognizerLanguagePack='en-ZA',
        AS_PER_CHAR_TIME_DURATION=0.075,
        culture='en-ZA',
        encoding='English_South Africa.20127'
where index1=15

Update RecognitionLanguages
set encoding='French_France.28591' where index1=5
GO
```

```
GO
Update RecognitionLanguages
set Display='',
    NuanceRecognizerLanguagePack=NULL,
        AS_PER_CHAR_TIME_DURATION=NULL,
        culture=NULL,
        DictionaryName=NULL,
        encoding=NULL
where index1=15


GO

GO
Alter FUNCTION [dbo].[sp_getTopicEventSequences]
(
@callid int, @threshold int
)
RETURNS varchar(max)
AS
BEGIN
declare @text as varchar(max)
set @text=''
select @text=@text + ltrim(rtrim(str(e.resourceID))) + ' '
from callEventstbl e
where e.callid = @callid
and e.type=1 and e.subtype=1
and e.confidence>=@threshold
and e.found=1
order by e.starttime
RETURN substring(@text, 1, len(@text))
END
go

GO
update dbo.callrecognizer set ADJUST_CONF = 'False'
GO

GO
Delete from wildcardGrammars
where [key]='[voiceenroll]'
```

```
GO
BEGIN TRANSACTION
SET QUOTED_IDENTIFIER ON
SET ARITHABORT ON
SET NUMERIC_ROUNDABORT OFF
SET CONCAT_NULL_YIELDS_NULL ON
SET ANSI_NULLS ON
SET ANSI_PADDING ON
SET ANSI_WARNINGS ON
COMMIT
BEGIN TRANSACTION
GO
EXECUTE sp_rename N'dbo.wildcardGrammars.[key]', N'Tmp_token',
'COLUMN'
GO
EXECUTE sp_rename N'dbo.wildcardGrammars.value', N'Tmp_grammar_1',
'COLUMN'
GO
EXECUTE sp_rename N'dbo.wildcardGrammars.Tmp_token', N'token',
'COLUMN'
GO
EXECUTE sp_rename N'dbo.wildcardGrammars.Tmp_grammar_1',
N'grammar', 'COLUMN'
GO
ALTER TABLE dbo.wildcardGrammars SET (LOCK_ESCALATION = TABLE)
GO
COMMIT

GO
declare @table_name nvarchar(256)
declare @col_name nvarchar(256)
declare @Command  nvarchar(1000)

set @table_name = N'ComputerList'
set @col_name = N'isBackupConfigurationServer'

select @Command = 'ALTER TABLE ' + @table_name + ' drop constraint
' + d.name
 from sys.tables t
  join    sys.default_constraints d
   on d.parent_object_id = t.object_id
  join    sys.columns c
```

```
   on c.object_id = t.object_id
    and c.column_id = d.parent_column_id
 where t.name = @table_name
  and c.name = @col_name

execute (@Command)
GO

IF EXISTS(select * from sys.columns where Name =
N'isBackupConfigurationServer' and Object_ID =
Object_ID(N'ComputerList'))
BEGIN
ALTER TABLE [dbo].[ComputerList] DROP COLUMN
isBackupConfigurationServer
END
GO


GO

ALTER PROCEDURE [dbo].[sp_deleteTexts](@TextIDs
varchar(max),@deletedTextsNum int OUTPUT)
         WITH EXECUTE AS SELF
AS
BEGIN
        SET NOCOUNT ON;
        DECLARE @ParmDefinition nvarchar(500);
        set @deletedTextsNum ='-1'
        declare @SelectCounter nvarchar(max)
        set @SelectCounter = N'select @sOUT = count(*) from
textData where textId in (' + @TextIDs + ')'
        SET @ParmDefinition = N'@sOUT int OUTPUT';
    exec sp_executesql
@SelectCounter,@ParmDefinition,@sOUT=@deletedTextsNum OUTPUT;
        declare @deleteTexts as nvarchar (max)
        set @deleteTexts = ''
        ---------
        set @deleteTexts = @deleteTexts + 'DELETE FROM
callMetaExTbl where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM
callcategoryTbl where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM
```

```
callEventsTbl where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM callAgentTbl
where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM
callPartitionTbl where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM textparties
where textId in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM textStatus
where textId in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM textData
where textId in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM
repCategoryMetaTbl where callid in (' + @TextIDs + ')'
        set @deleteTexts = @deleteTexts + 'DELETE FROM
callSpeakersTbl where callid in (' + @TextIDs + ')'
        ---------
        exec sp_executesql @deleteTexts
END


GO
ALTER TABLE callNotesTbl DROP CONSTRAINT FK_callNotesTbl_callMetaTbl

GO
UPDATE RecognitionLanguages
set DictionaryName = 'pt_BR' where index1=11
GO
UPDATE RecognitionLanguages
set DictionaryName = 'es_MX' where index1 = 1
GO
UPDATE RecognitionLanguages
set DictionaryName = 'es_ES' where index1 = 2
GO

GO
delete from reportCachingQueue
GO
alter table reportCachingQueue add runAtTime int NOT NULL
GO

CREATE PROCEDURE [dbo].[sp_getReportCachingNextTask]
AS
```

```
BEGIN

        BEGIN TRAN

        declare @id int
        SELECT top 1 @id=id from reportCachingQueue with(updlock
holdlock)
        where nextTimeToRun<dbo.time2tod(GetUtcDate())
        order by nextTimeToRun

        IF @id IS NOT NULL
        BEGIN
                update reportCachingQueue set
nextTimeToRun=dbo.time2tod(GetUtcDate())+1*60*60 where id=@id
                select reportid, [partitions], runAtTime from
reportCachingQueue where id=@id
        END
        COMMIT

END
GO

grant execute on sp_getReportCachingNextTask to platform
GO


GO

BEGIN TRANSACTION
SET QUOTED_IDENTIFIER ON
SET ARITHABORT ON
SET NUMERIC_ROUNDABORT OFF
SET CONCAT_NULL_YIELDS_NULL ON
SET ANSI_NULLS ON
SET ANSI_PADDING ON
SET ANSI_WARNINGS ON
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.indexParamsTbl ADD
        autoRestore bit NOT NULL CONSTRAINT
DF_indexParamsTbl_autoRestore DEFAULT 0,
```

```
        timeToWaitForFolder int NULL
GO
ALTER TABLE dbo.indexParamsTbl SET (LOCK_ESCALATION = TABLE)
GO
COMMIT

GRANT update(indexdirectorypath, dailyBackupLocation) ON
[indexparamstbl]  TO [Platform]
GO

GO
grant update(status,stoppedForPurge) on computerList to
interactionReceiver
grant execute on sp_enqueuePurgeTask to interactionReceiver



GO
ALTER PROCEDURE [dbo].[sp_getClusterTrendData]
        @clusterId int,
        @maxTrendWeeksBack int
AS
BEGIN

        declare @firstDayOfWeek as int
        set @firstDayOfWeek = dbo.getFirstDayOfWeek()
        set DATEFIRST @firstDayOfWeek

        declare @paramsId as int
        declare @reportDate as datetime

        set @paramsId = (select paramsId from ExplorationReports
where reportid=(select reportid from ExplorationReportsData where
clusterid=@clusterId))
        set @reportDate = (select dbo.tod2time(endTime) from
ExplorationReportParams where id=@paramsId)

        IF @reportDate IS NULL
        BEGIN
                select NULL as[time], NULL as [numberOfCalls], NULL
as [totalNumberOfCalls]
                RETURN -- the specified cluster id does not exist.
```

```
        END

        create table #trendData (
                [time] datetime NOT NULL,
                [numberOfCalls] int,
                [totalNumberOfCalls] int
        )

        insert into #trendData
        select dbo.tod2time([time]) as [time], numberOfCalls,
totalNumberOfCalls
        from dbo.ExplorationTrendData where clusterid=@clusterId

        IF NOT EXISTS(select [time] from #trendData)
        BEGIN
                insert into #trendData values(DATEADD(DAY,
1-DATEPART(WEEKDAY, @reportDate), @reportDate), NULL, NULL)
        END

        -- add the dates (start of week) from the last date with
data to the report time
        while dateadd(week, 1, (select max(time) from #trendData))
< @reportDate
        BEGIN
                declare @newTime as datetime
                set @newTime = dateadd(week, 1, (select max(time)
from #trendData))
                IF not exists(select * from #trendData where [time]
= @newTime)
                BEGIN
                        insert into #trendData values(@newTime,
NULL, NULL)
                END
        END

        declare @currentTrendDate as datetime
        set @currentTrendDate = (select max(time) from #trendData)

        -- skip the first week if the current report date is the
first day of the week
        if @currentTrendDate = @reportDate
        BEGIN
```

```
                set @currentTrendDate = dateadd(week, -1,
@currentTrendDate)
        END

        declare @count as int
        set @count = 0
        while @count < @maxTrendWeeksBack - 1
        BEGIN
                IF not exists(select * from #trendData where [time]
= @currentTrendDate)
                BEGIN
                        insert into #trendData
values(@currentTrendDate, NULL, NULL)
                END

                set @currentTrendDate = dateadd(week, -1,
@currentTrendDate)
                set @count = @count + 1
        END


        set @currentTrendDate = dateadd(week, 1, @currentTrendDate)

        delete from #trendData where time<@currentTrendDate --
remove the old items if there are more than @maxTrendWeeksBack items

        select * from #trendData order by [time]

        drop table #trendData

END


GO
GO

GRANT UPDATE ON dbo.[TextStatus] TO Web

GO

GO
```

```
BEGIN TRANSACTION
SET QUOTED_IDENTIFIER ON
SET ARITHABORT ON
SET NUMERIC_ROUNDABORT OFF
SET CONCAT_NULL_YIELDS_NULL ON
SET ANSI_NULLS ON
SET ANSI_PADDING ON
SET ANSI_WARNINGS ON
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.TextData
        DROP CONSTRAINT FK_TextData_ResourceType
GO
ALTER TABLE dbo.ResourceType SET (LOCK_ESCALATION = TABLE)
GO
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.TextData
        DROP CONSTRAINT FK_TextData_programInfoTbl
GO
ALTER TABLE dbo.programInfoTbl SET (LOCK_ESCALATION = TABLE)
GO
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.TextData
        DROP CONSTRAINT DF_TextData_customerId
GO
ALTER TABLE dbo.TextData
        DROP CONSTRAINT DF_TextData_customerGroupId
GO
ALTER TABLE dbo.TextData
        DROP CONSTRAINT DF_TextData_programId
GO
CREATE TABLE dbo.Tmp_TextData
        (
        textId int NOT NULL,
        resourceTypeId int NOT NULL,
        originalTime int NULL,
        customerId varchar(256) NOT NULL,
```

```
        customerGroupId varchar(256) NOT NULL,
        subject nvarchar(MAX) NULL,
        body nvarchar(MAX) NULL,
        externalTextId varchar(256) NULL,
        sender varchar(8000) NULL,
        receiver varchar(8000) NULL,
        cc varchar(8000) NULL,
        bcc varchar(8000) NULL,
        programId int NOT NULL,
        originalBody varchar(MAX) NULL
        )  ON [PRIMARY]
         TEXTIMAGE_ON [PRIMARY]
GO
ALTER TABLE dbo.Tmp_TextData SET (LOCK_ESCALATION = TABLE)
GO
GRANT DELETE ON dbo.Tmp_TextData TO Platform  AS dbo
GO
GRANT INSERT ON dbo.Tmp_TextData TO Platform  AS dbo
GO
GRANT SELECT ON dbo.Tmp_TextData TO Web  AS dbo
GO
GRANT SELECT ON dbo.Tmp_TextData TO Platform  AS dbo
GO
GRANT SELECT ON dbo.Tmp_TextData TO SMART  AS dbo
GO
GRANT UPDATE ON dbo.Tmp_TextData TO Web  AS dbo
GO
GRANT UPDATE ON dbo.Tmp_TextData TO Platform  AS dbo
GO
ALTER TABLE dbo.Tmp_TextData ADD CONSTRAINT
        DF_TextData_customerId DEFAULT ('') FOR customerId
GO
ALTER TABLE dbo.Tmp_TextData ADD CONSTRAINT
        DF_TextData_customerGroupId DEFAULT ('') FOR customerGroupId
GO
ALTER TABLE dbo.Tmp_TextData ADD CONSTRAINT
        DF_TextData_programId DEFAULT ((0)) FOR programId
GO
IF EXISTS(SELECT * FROM dbo.TextData)
         EXEC('INSERT INTO dbo.Tmp_TextData (textId,
resourceTypeId, originalTime, customerId, customerGroupId, subject,
body, externalTextId, sender, receiver, cc, bcc, programId,
```

```
originalBody)
                SELECT textId, resourceTypeId, originalTime,
customerId, customerGroupId, CONVERT(nvarchar(MAX), subject),
CONVERT(nvarchar(MAX), body), externalTextId, sender, receiver, cc,
bcc, programId, originalBody FROM dbo.TextData WITH (HOLDLOCK
TABLOCKX)')
GO
ALTER TABLE dbo.TextStatus
        DROP CONSTRAINT FK_TextStatus_TextData
GO
DROP TABLE dbo.TextData
GO
EXECUTE sp_rename N'dbo.Tmp_TextData', N'TextData', 'OBJECT'
GO
ALTER TABLE dbo.TextData ADD CONSTRAINT
        PK_TextDataTbl PRIMARY KEY CLUSTERED
        (
        textId
        ) WITH( STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]

GO
CREATE NONCLUSTERED INDEX IX_externalTextId ON dbo.TextData
        (
        externalTextId
        ) WITH( STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
GO
ALTER TABLE dbo.TextData ADD CONSTRAINT
        FK_TextData_programInfoTbl FOREIGN KEY
        (
        programId
        ) REFERENCES dbo.programInfoTbl
        (
        programId
        ) ON UPDATE  NO ACTION
         ON DELETE  NO ACTION

GO
ALTER TABLE dbo.TextData ADD CONSTRAINT
        FK_TextData_ResourceType FOREIGN KEY
        (
```

```
        resourceTypeId
        ) REFERENCES dbo.ResourceType
        (
        resourceTypeId
        ) ON UPDATE  NO ACTION
         ON DELETE  NO ACTION

GO
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.TextStatus WITH NOCHECK ADD CONSTRAINT
        FK_TextStatus_TextData FOREIGN KEY
        (
        textId
        ) REFERENCES dbo.TextData
        (
        textId
        ) ON UPDATE  NO ACTION
         ON DELETE  NO ACTION

GO
ALTER TABLE dbo.TextStatus
        NOCHECK CONSTRAINT FK_TextStatus_TextData
GO
ALTER TABLE dbo.TextStatus SET (LOCK_ESCALATION = TABLE)
GO
COMMIT


GO
UPDATE [webServiceParams] SET
[sendFeedbackEmail]='customercare@genesys.com'



GO
delete wildcardGrammars where grammar='digits'

GO
GO
```

```
UPDATE indexer set calls_per_chunk=6000

GO

GO
update dbo.versionTbl set version= '8.5.7104' where resource in
('SM', 'SMART')
go
```

SpeechMiner 8.5.0.1 to 8.5.2 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.5.0.1 to version 8.5.2.

Pre-upgrade Requirements

- Request the most recent release of the 8.5 software from your Genesys representative.
- Request the SpeechMiner license from Genesys Licensing.
- Verify that the following are installed:
  - Microsoft .NET Framework 4.5 SP1 (4.5.1) must be installed on all machines that will run SpeechMiner components or interact with SpeechMiner.
    You can download the installation package at:
    http://www.microsoft.com/en-us/download/details.aspx?id=40773.
  - Microsoft Visual C++ 2013 Redistributable must be installed on all machines that will run SpeechMiner components or interact with SpeechMiner.
    You can download the installation package at:
    http://www.microsoft.com/en-us/download/details.aspx?id=40784.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Storage Requirements	To successfully complete the upgrade process, the data partition in the SQL server must have available disk space. The minumum required storage for the upgrade should be twice the size of the production database .mdf file.
Perform Pre-upgrade Tests	Since upgrading SpeechMiner can take several hours it is important to avoid delays. For this reason, it is recommended to test SMUpgrade before you begin the upgrade procedure. To test SMUpgrade, perform one of the following:

	• Create a copy of the database and send it to your Genesys counterpart. Genesys will test the SMUpgrade process in a lab environment. • Create a copy of the database on a separate SQL server. Provide Genesys with the details of the SQL server and Genesys will test the SMUpgarde process on this temporary server.
Check for Customization	If any customizations were implemented on your DB, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.
Purging Old Data	Most systems have a data retention policy in place. Data (for example, audio,exploration data, etc.) that is older than the specified period of time is automatically deleted. If you do not have a data retention policy in place, it is recommended to determine what data should be saved and what can be discarded, since deleting the data before the upgrade will reduce both the time it takes to run the upgrade process and the storage-space requirements.
Compressed Audio Format	If your previous system stored a compressed format different from an MP3 format you should perform one of the following: • When your system saves WAV PCM recognition files convert the files to MP3 using on-the-fly conversion. • When your system does not save WAV PCM recognition files convert the audio files to MP3 audio files using an external process and update the file names in the database accordingly (field 'filename' in the table 'callaudiotbl').

Rollback Plan

To ensure that you can revert back to SpeechMiner 8.5, keep the 8.5 DB active on the server, rather than just keeping a backup file.

Do not delete the 8.5 data folders (`index`, `grammars`, etc.). Instead, configure the 8.5 system with new data folders. Create the following new folders to ensure that you will not lose 8.5 data:

- Create the following empty folders:
  - Input
  - Interaction Receiver Input
  - Filtered
  - Index (this folder will be populated during the upgrade procedure).

- Copy the content of the following existing folders to new folders with the same name:
  - Store
  - Grammar
  - Backup

For detailed information about the folders you should create, refer to Required Folders

Since the 8.5 DB and data folders are saved and available, back-out steps are not required if problems arise with the upgrade process before you uninstall 8.5. The 8.5 system should still be configured and functional.

After you uninstall SpeechMiner 8.5 and install SpeechMiner 8.5.2, the only way to revert back to 8.5 is to install 8.5 again and update the config files using SMConfig. However, since the DB and data folders would not have been deleted, they should be available and ready to use without changing the system configuration.

Upgrade Procedure

The following table lists the approximate times required to complete the upgrade steps:

Step	Time
Stop the system (step 1)	15 minutes
Backup the database (step 2)	120 minutes
Create target database (step 3)	30 minutes
Run SMUpgrade (steps 6 to 13)	10 to 20 hours
Configuring and starting the system (steps 14 to end)	60 minutes

1. Using `SMConfig->Services->Stop Services`, stop the 8.5 system.
2. Create a copy of the source DB and upgrade it to the latest build:
   The source DB must be in build 8.5.7104. Refer to the `versionTbl` table to determine the correct version.
   If you have a build that is later than 8.5.7104, contact Genesys Customer Care.
   If the source DB is not the latest build and you do not want to update it to the latest build, create a copy of the source DB and update the copy to the latest build.

   Important

These steps are necessary because the 8.5 DB schema needs to be updated to the latest schema in order for the rest of the upgrade process to succeed.

Use the copy of the source DB as the baseline for the 8.5.2 upgrade.

a. Back up the 8.5 index folder to a backup folder (see Configuring SpeechMiner—Index).
b. Create a copy of the source DB (back up the DB and then restore it in another location).
c. Configure the copy of the DB with the Index backup. This will be the baseline for the upgrade.
d. Implement the schema changes on the baseline DB to bring the schema into line with the latest 8.5 build version. For this step you will need assistance from Customer Care.

3. Create the 8.5.2 target DB as follows:
   - Manually—Refer to **Installing the SpeechMiner Database > Manual** tab.

   Or
   - Setup Wizard— Refer to **Installing the SpeechMiner Database > Setup Wizard** tab.
4. If the MS-SQL server is an Enterprise Edition, run `EXEC sp_create_DB_storage_partitions` on the target database.
5. If your source and target databases are on different servers, make sure the servers are linked in both directions, using the stored procedures `sp_addlinkedserver` and `sp_addlinkedsrvlogin`, as needed.
6. Install and run SMUpgrade (to migrate the data from the 8.5 DB to the 8.5.2 DB), as follows:

Prerequisites:

- When migrating a large database, make sure that the hard drive that hosts the target database has enough storage space.

Usage

a. Query the `versionTbl` table to ensure that your 8.5 source database is updated to the latest 8.5 schema.
b. Verify that your recovery model is either `Simple` or `Bulk-logged`. To determine which recovery model you have, right click `db > properties > options > recovery model`.
c. Use the SpeechMiner Installer to install the `SMUpgrade` component. It is recommended to install and run the SMUpgrade component on the SQL server.
d. Configure the following in the `\utopy\tools\bin\release\SMUpgrade.exe.config` file:
    - file locations
    - tables to skip (comma separated list)
    - number of threads running concurrently on a large table
    - bulk copy usage

    It is recommended that the `bulk-load` folder be located on the SQL server. The `bulk-load` folder must be shared and the user running the SQL server service must have full control over it.
    If you do not set the bulk copy usage, the `callAudioTbl` upgrade will take up to two times longer on a large DB.
    Only use the `skip-tables` configuration if specifically requested by Genesys Customer Care.

```
 <appSettings>
<add key="ErrorLogFile"
value=".\SMUpgradeLog.txt" />
<add key="LogFile"
value=".\tableLog.txt" >
<add key="TimingsFile"
value=".\tableTimings.txt" />
<add key="SkipTables"
value="callTasksTbl" />
```

```
<add key="NumThreads" value="10" />

</appSettings>
```

e. Run `SMUpgrade.exe`.
Log in and select the appropriate 8.5 source and 8.5.2 destination databases.
The databases that appear in the old databases drop down list include ver8_5 in their file name. The databases in the new databases drop down list, include ver8_5_2 in their file name. You can also type relevant databases that are named differently.

> **Important**
>
> It is highly recommended to use the `sa` credentials or a user account with bulk insert permissions.
>
> The user account must belong to the `db_owner` role in the target database. By default, the `DBUser` does not include the `db_owner` role.

f. The GUI shades the tables as follows:
   - Green—The table is finished. Both the source and the target DBs contain the same number of records. This conclusively shows that either the data has been copied or there is no data to copy.
   - Yellow—The number of records in the source and target DBs is not indicative of whether the

data in both DBs is identical or not. It is therefore not known whether the table is finished or not.
- Red—The table is not finished. The number of records in the source and target databases are not the same, and indicating that the data has not been copied in full.

g. Click `Full Upgrade` to run the upgrade, or `Resume Last` if your previous upgrade was interrupted.

   Resuming the last upgrade will shorten the time to run, but might cause problems. To avoid such problems, restore the database again and run the full upgrade.
   You can also define tables to skip in the configuration file. Every step in the upgrade process is shown in the GUI.
   You can stop the upgrade by clicking the `Close` button. You will be prompted to confirm your action. Note that the window closes immediately, but the process still runs for a while, as it needs to re-enable the indexes it disables when it starts running.
   The time each step took is written to the `TimingsFile`. The location of this file is defined in the configuration file.
   IN CASE OF FAILURE: Review all status, error and exception notifications in the `ErrorLogFile`.

h. Continue with the upgrade instructions below.

7. If the SpeechMiner Maintenance Job exists, and the `Update time table` step is included, delete the `Update time table` step. Make sure the last step in the job is set to quit the job upon both success and failure.
8. Run SQL commands on the new DB to update audio formats:

```
insert into audioFormatsTbl values(3,'MP3',1,'mp3',44100)
declare @pcm2mp3 int
declare @mp32pcm int
declare @mp32mp3 int
insert into audioConversionTypesTbl values
(1,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_PCM to MP3')
select @pcm2mp3=@@IDENTITY
insert into audioConversionTypesTbl values
(2,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'VOX to MP3')
insert into audioConversionTypesTbl values
(3,3,4,null,null,8000,'MP3 to MP3')
set @mp32mp3=@@IDENTITY
insert into audioConversionTypesTbl values
(4,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_ADPCM to
MP3')
insert into audioConversionTypesTbl values
(5,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_TRUESPEECH
to MP3')
insert into audioConversionTypesTbl values
(6,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_GSM610 to
MP3')
insert into audioConversionTypesTbl values
(8,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_MULAW to
MP3')
insert into audioConversionTypesTbl values
(9,3,1,'ffmpeg.exe','-i {0} -f mp3 {1}',8000,'WAV_ALAW to MP3')
insert into audioConversionTypesTbl values
(3,1,1,'ffmpeg.exe','-i {0} {1}',8000,'MP3 to WAV_PCM')
select @mp32pcm=@@IDENTITY
update audioConversionGroupsTbl set
audioConversionTypeID=@mp32pcm where
audioConversionTypeID in (select audioConversionTypeId from
audioConversionTypesTbl where fromFormat=6 and
toFormat=1)
update audioConversionGroupsTbl set
audioConversionTypeID=@pcm2mp3 where
audioConversionTypeID in (select audioConversionTypeId from
```

```
audioConversionTypesTbl where fromFormat=1 and
toFormat=6)
update audioConversionGroupsTbl set
audioConversionTypeID=@mp32mp3 where
audioConversionTypeID in (select audioConversionTypeId from
audioConversionTypesTbl where fromFormat=6 and
toFormat=6)
update audioConversionGroupsTbl set
description=replace(description,'WAV_GSM610','MP3')
update audioFormatsTbl set uplatformSupported=0 where
audioFormatId not in (0,1,3,7)
delete audioConversionGroupsTbl where
audioConversionTypeID in (select audioConversionTypeId from
audioConversionTypesTbl where toFormat not in (select
audioFormatId from audioFormatsTbl where
uplatformSupported=1))
delete audioConversionTypesTbl where toFormat not in (select
audioFormatId from audioFormatsTbl where
uplatformSupported=1)
update siteAudioFormatsTbl set audioFormatId=3 where
audioFormatId=6
update audioFormatsTbl set uplatformSupported=1
```

9. Optional: Uninstall 8.5 from all servers. The two versions (8.5 and 8.5.2) cannot be running side by side at the same time. Only one version can be registered as the active SpeechMiner service on each server. The installation binaries can be left on the server.
10. Update Microsoft .NET Framework.
11. Update Microsoft C++ Redistributable.
12. Install the 8.5.2 platform on all servers.
13. Install 8.5.2 Web on the Web server.
14. Install 8.5.2 SMART on users' desktops, as required.
15. Deploy SQLCLR on the DB server.
16. Update the package folders with the 8.5.2 `.gram` files. The `.gram` files are located in the `<Installation Folder>/Support/Grammars`.

Alternatively, if you have not made changes to any file in these folders, you can delete their content completely. SMConfig will copy the grammar files to `<Installation folder>/Support/Grammars`.

17. Manually copy the files in `<Installation Folder>\Support\Grammars\Confidence` to the `Global Packages` folder.
18. Copy the following commands and paste them into the **New Query** text area:

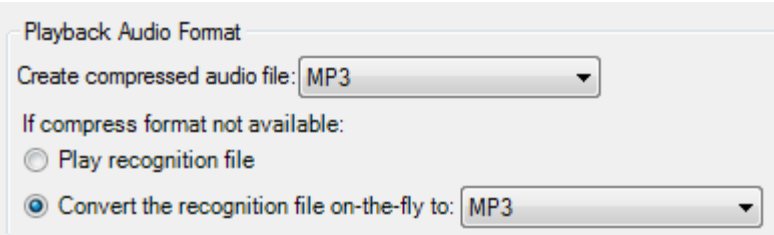
```
EXEC sp_configure 'xp_cmdshell', 1
GO
```

19. Run `SMConfig`.

> **Important**
>
> If your target database was restored from a backup file, you may need to "fix" an orphan dbuser. To do this, simply run EXEC sp_change_users_login 'Auto_Fix', 'dbuser'.

    a. Configure the Sites & Machines, panel as necessary, and save the changes. Make sure you save this panel even if you have not made any changes.
    b. Configure the Services panel and save the changes. Do not start any of the services.
    c. Configure the Index panel and save the changes.
    d. Update the SpeechMiner License with the new 8.5.2 licenses provided by Genesys Licensing.
    e. In the Reports panel, update the `MRSLibrary.dll` on the report server.
    f. Deploy the reports to the report server.
    g. In the **Audio** panel (when using compression):
        - Change the compressed format to **MP3**.
        - Under **If compress format not available** select **Convert the recognition file on-the-fly to MP3**.

    The player no longer supports compressed formats other than MP3. On-the-fly conversion is required to listen to previously compressed audios.
    For additional information refer to the Audio section in the Administration Guide.



h. Run the following query to convert GSM files to MP3:
Replace WAV_GSM610 with your current audio format (see AudioFormatsTbl).

```
declare @formatName varchar(40)
set @formatName='WAV_GSM610'
declare @formatId int
select @formatId=audioFormatId from audioFormatsTbl
where audioFormatName = @formatName
declare @conversionType int
declare @conversionDesc varchar(100)
select @conversionType=
audioConversionTypeId,@conversionDesc=Description
from audioConversionTypesTbl where fromFormat=
@formatId and toFormat=3
insert into audioConversionGroupsTbl
values(6,@conversionType,3,'PLAYER - ' +
@conversionDesc)
```

20. Start SMART and perform the following:
    a. Right-click on each active Program icon and choose `Activate program`. This will mark all the Programs, Topics, and Categories as changed.
    b. Click the `Apply` button.
    c. In the new `Apply` popup window, choose `Apply all`.
    d. Click the `Apply` button.
21. Using `SMConfig`, start the `UPlatform` services on all the servers.
22. Update the Stored Procedures by copying any existing custom Stored Procedures from the 8.5 DB to the 8.5.2 DB.
    It is not necessary to copy Stored Procedures that are used by gauges, and are in the `GaugeWidgetProcedures` table, because they are copied automatically.

23. Open the SpeechMiner web-based interface and test the functionality.
24. Update the Database Jobs:
    - All database jobs that point to the 8.5 DB should be changed to point to the new 8.5.2 DB. Examples of DB jobs that might need to be changed:
        - DB maintenance job
        - `sp_agentFilterCleanByDays`
        - `sp_updateUntilYesterdayMaxChannels`

    To change a DB job, we recommend that you edit the Job Step property using the SQL Management studio.
25. In the SpeechMiner web-based interface, manually reschedule 8.5 reports that should continue to run on a scheduled basis.
26. If one or more of your users changed their default SpeechMiner homepage in the previous version, update the url format to the selected page in the new version.

SpeechMiner 8.0 or 8.1 to 8.5.2 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.0 or 8.1 to version 8.5.2. Upgrading from a previous version to 8.5.2 enables you to retain your existing data.

Upgrade from 8.0 to 8.5.2

1. Upgrade from 8.0 to 8.1.
2. Upgrade from 8.1 to 8.5.
3. Upgrade from 8.5 to 8.5.2.

Upgrade from 8.1 to 8.5.2

1. Upgrade from 8.1 to 8.5.
2. Upgrade from 8.5 to 8.5.2.

Important

Deploying a new SpeechMiner 8.5.2 installation does not require performing the upgrade procedures. However, a new installation will not enable you to retain existing data. For additional information about installing SpeechMiner 8.5.2, refer to the *Administration Guide*.

SpeechMiner 8.5.2 to 8.5.201 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.5.2 to version 8.5.201

Pre-upgrade Requirements

- Request the newest 8.5.201 software from Genesys.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Check for Customization	If any customizations were implemented on your database, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.

Time Requirements

The following table lists the approximate times required to complete the upgrade steps:

Step	Time
Stop the system (step 1)	10 minutes
Backup the database (step 3)	15 minutes (the larger the database the more time this step will take).
Configuring and starting the system (steps 14 to end)	40 minutes

Upgrade Procedure

1. Close all browsers and SpeechMiner applications.
2. Stop the **Uplatform** service.
3. Backup the SpeechMiner 8.5.2 database in the SQL server.
4. Uninstall SpeechMiner 8.5.2 (build 194).

5. If SpeechMiner applications in IIS are not removed, remove the Interaction Receiver and SpeechMiner Web application manually.
6. Copy the entire **FullInstaller** folder from the 8.5.201 build 252 kit to your local server.
7. Install the new SpeechMiner 8.5.201 (build 252).
8. Provide the relevant user with IIS_IUSRS group read/write/modify (that is, Windows file Security) permissions for the new SpeechMiner Installation folder.
9. Reboot your machine.
10. Perform SpeechMiner 8.5.2 database changes on the SQL Manager by running all the SpeechMiner 8.5.2 database commands in the SQL query window.
    To receive the relevant commands, see Database Changes - Commands.
11. Deploy SQLCLR: Via SQL management run the commands that are in C:\Program Files (x86)\Genesys\Software\Support\sqlclr.sql in the SpeechMiner 8.5.2 database.

    For details refer to the SpeechMiner Administration Guide (Installing the SpeechMiner Components > Installing the SpeechMiner Database > SQL CLR).

12. Run SMConfig and connect to the database that was upgraded.
13. Click **Save** in **Sites & Machines**.
14. Deploy the MRSLibrary and all the Reports in the Reports tab.
15. In the **SpeechMiner Configuration Tool 8.5.2 > Services** select the following under the **Services** tab:
    - Create Performance Counters
    - Register services
    - Update config files
    - Encrypt config files
    - Select/Deselect All > select the relevant machine.
    - Restart Services > change status to run.
    - Under **Credential** enter the relevant user's information.
    - Click **Save**.
16. Open the SpeechMiner Web and check its functionality.

8.5.2 to 8.5.201 Database Changes - Commands

Copy the following commands and run them in the SQL query window for the SpeechMiner 8.5.2 to 8.5.201 Upgrade Procedure:

```
INSERT INTO [dbo].[objectPermissionsTbl]
           ([objectId]
```

```
           ,[groupId]
           ,[description]
           ,[values]
           ,[configurable]
           ,[explanation])
     VALUES
           (71
           ,14
           ,'View All Reports'
           ,'<Values xmlns:xsd="http://www.w3.org/2001/
XMLSchema"><Value><display>View All
Reports</display><value>2</value></Value><Value><display>View Only
My Reports And Shared
Reports</display><value>0</value></Value></Values>'
           ,1
           ,'')
GO


          GO

GO

INSERT INTO [dbo].[rolePermissionsTbl]
           ([objectId]
           ,[role]
           ,[permission])
     VALUES
           (71
           ,5
           ,2)
GO


          GO

GRANT UPDATE ON [dbo].[configServer] TO SMConfig  AS dbo
          GO

GO

INSERT INTO [dbo].[objectPermissionsTbl]
```

```
           ([objectId]
           ,[groupId]
           ,[description]
           ,[values]
           ,[configurable]
                  ,[explanation])
     VALUES
           (711
           ,41
           ,'Recording Panel'
           ,'<Values xmlns:xsd="http://www.w3.org/2001/
XMLSchema"><Value><display>Show</display><value>2</value></Value><Value><disp
           ,1
                  ,'')
GO


          GO

GO

ALTER TABLE [dbo].[recentSMConfigApply] DROP CONSTRAINT
[DF__recentSMC__textA__77FFC2B3]
GO

/****** Object:  Table [dbo].[recentSMConfigApply]    Script Date:
18/08/2014 14:19:22 ******/
DROP TABLE [dbo].[recentSMConfigApply]
GO

/****** Object:  Table [dbo].[recentSMConfigApply]    Script Date:
18/08/2014 14:19:22 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[recentSMConfigApply](
        [sitesApply] [int] NOT NULL,
        [reportsApply] [int] NOT NULL,
        [licenseApply] [int] NOT NULL,
```

```
        [servicesApply] [int] NOT NULL,
        [audioManagerApply] [int] NOT NULL,
        [indexApply] [int] NOT NULL,
        [textAnalyticsApply] [int] NOT NULL,
        [recordingParametersApply] [int] NOT NULL
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[recentSMConfigApply] ADD  CONSTRAINT
[DF__recentSMC__textA__77FFC2B3]  DEFAULT ((0)) FOR
[textAnalyticsApply]
GO

INSERT INTO [dbo].[recentSMConfigApply]
           ([sitesApply]
           ,[reportsApply]
           ,[licenseApply]
           ,[servicesApply]
           ,[audioManagerApply]
           ,[indexApply]
           ,[textAnalyticsApply]
           ,[recordingParametersApply])
     VALUES
           (0,
            0,
            0,
            0,
            0,
            0,
            0,
            0)
GO

GRANT SELECT ON [dbo].[recentSMConfigApply] TO SMConfig  AS dbo
GRANT UPDATE ON [dbo].[recentSMConfigApply] TO SMConfig  AS dbo


          GO

GO
```

```
INSERT INTO [dbo].[rolePermissionsTbl]
           ([objectId]
           ,[role]
           ,[permission])
     VALUES
           (711
           ,5
           ,2)
GO


          GO

update RecognitionLanguages
set Display = 'Japanese', NuanceRecognizerLanguagePack = 'ja-JP',
MinIndexConfidence = 40, culture='ja-JP', DictionaryName = 'ja_JP'
where index1 = 17
update RecognitionLanguages set SetWordAsKeywordLength = 2 where
index1 = 17
update RecognitionLanguages set AS_PER_CHAR_TIME_DURATION = 0.25
where index1 = 17
GO

delete RecognitionParams where language=17
insert into RecognitionParams values(17, 5, 'swirec_lmweight',
'0.4')
insert into RecognitionParams values(17, 5, 'swirec_max_arcs',
'3300')
insert into RecognitionParams values(17, 5, 'swirec_word_penalty',
'0.1')
GO

delete LVCSRGrammarParams where language=17
insert into LVCSRGrammarParams values(17, 'ngram_order', '3')
insert into LVCSRGrammarParams values(17, 'cutoffs', '1 1')
GO

delete WordconfidenceInterpolation where language=17
insert into WordConfidenceInterpolation values(17, '-0.03 0.7',
'0.04 0.35', '0.03 0.3')
update WordConfidenceInterpolation set wer = '-0.0757 0.8914' where
language = 17
```

```
update WordConfidenceInterpolation set precision = '0.0486 0.3022'
where language = 17
update WordConfidenceInterpolation set recall = '0.0735 0.1606'
where language = 17
GO

delete stopwords where language=17
insert into stopWords
values(17,N'??,??,??,??,??,??,???,?,??,???,??,?,?,??,??,?,??,??,?,?,???,??,??
'',''');
GO
          GO

Update RecognitionLanguages
set MinIndexConfidence = 45 where index1 = 11
GO
Update RecognitionLanguages
set MinIndexConfidence = 60 where index1 = 14
GO
          GO

delete wildcardGrammars where languageId=17
insert into wildcardGrammars values(17, '[*]','[*]',1.5,null)
insert into wildcardGrammars values(17,
'[alphanum]','alphanum',1,null)
insert into wildcardGrammars values(17,
'[boolean]','boolean',1.5,null)
insert into wildcardGrammars values(17,
'[creditcard]','creditcard',3.5,null)
insert into wildcardGrammars values(17,
'[currency]','currency',2,null)
insert into wildcardGrammars values(17, '[date]','date',2,null)
insert into wildcardGrammars values(17, '[number]','number',1,null)
insert into wildcardGrammars values(17, '[phone]','phone',3.5,null)
insert into wildcardGrammars values(17,
'[postcode]','postcode',1.5,null)
insert into wildcardGrammars values(17, '[time]','time',1.5,null)
GO
          GO

delete wordPronunciationKeys where language=17
insert into wordPronunciationKeys values('b',17,1)
```

```
insert into wordPronunciationKeys values('by',17,2)
insert into wordPronunciationKeys values('d',17,3)
insert into wordPronunciationKeys values('dy',17,4)
insert into wordPronunciationKeys values('g',17,5)
insert into wordPronunciationKeys values('gy',17,6)
insert into wordPronunciationKeys values('k',17,7)
insert into wordPronunciationKeys values('kk',17,8)
insert into wordPronunciationKeys values('ky',17,9)
insert into wordPronunciationKeys values('p',17,10)
insert into wordPronunciationKeys values('pp',17,11)
insert into wordPronunciationKeys values('py',17,12)
insert into wordPronunciationKeys values('t',17,13)
insert into wordPronunciationKeys values('tt',17,14)
insert into wordPronunciationKeys values('f',17,15)
insert into wordPronunciationKeys values('h',17,16)
insert into wordPronunciationKeys values('hy',17,17)
insert into wordPronunciationKeys values('s',17,18)
insert into wordPronunciationKeys values('S',17,19)
insert into wordPronunciationKeys values('ss',17,20)
insert into wordPronunciationKeys values('SS',17,21)
insert into wordPronunciationKeys values('z',17,22)
insert into wordPronunciationKeys values('ts',17,23)
insert into wordPronunciationKeys values('tS',17,24)
insert into wordPronunciationKeys values('dZ',17,25)
insert into wordPronunciationKeys values('m',17,26)
insert into wordPronunciationKeys values('my',17,27)
insert into wordPronunciationKeys values('n',17,28)
insert into wordPronunciationKeys values('ny',17,29)
insert into wordPronunciationKeys values('N',17,30)
insert into wordPronunciationKeys values('4',17,31)
insert into wordPronunciationKeys values('4y',17,32)
insert into wordPronunciationKeys values('y',17,33)
insert into wordPronunciationKeys values('w',17,34)
insert into wordPronunciationKeys values('a',17,35)
insert into wordPronunciationKeys values('aa',17,36)
insert into wordPronunciationKeys values('E',17,37)
insert into wordPronunciationKeys values('I',17,38)
insert into wordPronunciationKeys values('II',17,39)
insert into wordPronunciationKeys values('M',17,40)
insert into wordPronunciationKeys values('MM',17,41)
insert into wordPronunciationKeys values('O',17,42)
insert into wordPronunciationKeys values('OO',17,42)
```

```
GO
          GO

delete Wordconfidenceinterpolation where language=17
insert into WordConfidenceInterpolation values(17, '-0.03 0.7',
'0.04 0.35', '0.03 0.3')
update WordConfidenceInterpolation set wer = '-0.0467 0.7316' where
language = 17
update WordConfidenceInterpolation set precision = '0.0337 0.5312'
where language = 17
update WordConfidenceInterpolation set recall = '0.0411 0.3426'
where language = 17
GO
          GO

/* To prevent any potential data loss issues, you should review
this script in detail before running it outside the context of the
database designer.*/
BEGIN TRANSACTION
SET QUOTED_IDENTIFIER ON
SET ARITHABORT ON
SET NUMERIC_ROUNDABORT OFF
SET CONCAT_NULL_YIELDS_NULL ON
SET ANSI_NULLS ON
SET ANSI_PADDING ON
SET ANSI_WARNINGS ON
COMMIT
BEGIN TRANSACTION
GO
ALTER TABLE dbo.LVCSRTrainingMaterial
        DROP CONSTRAINT DF_LVCSRTrainingMaterial_source
GO
CREATE TABLE dbo.Tmp_LVCSRTrainingMaterial
        (
        weight decimal(18, 2) NOT NULL,
        language int NOT NULL,
        sentence nvarchar(MAX) NOT NULL,
        source int NOT NULL
        )  ON [PRIMARY]
         TEXTIMAGE_ON [PRIMARY]
GO
ALTER TABLE dbo.Tmp_LVCSRTrainingMaterial SET (LOCK_ESCALATION =
```

```
TABLE)
GO
GRANT DELETE ON dbo.Tmp_LVCSRTrainingMaterial TO Platform  AS dbo
GO
GRANT INSERT ON dbo.Tmp_LVCSRTrainingMaterial TO Platform  AS dbo
GO
GRANT SELECT ON dbo.Tmp_LVCSRTrainingMaterial TO Platform  AS dbo
GO
GRANT SELECT ON dbo.Tmp_LVCSRTrainingMaterial TO SMART  AS dbo
GO
ALTER TABLE dbo.Tmp_LVCSRTrainingMaterial ADD CONSTRAINT
        DF_LVCSRTrainingMaterial_source DEFAULT ((0)) FOR source
GO
IF EXISTS(SELECT * FROM dbo.LVCSRTrainingMaterial)
         EXEC('INSERT INTO dbo.Tmp_LVCSRTrainingMaterial (weight,
language, sentence, source)
                SELECT weight, language, CONVERT(nvarchar(MAX),
sentence), source FROM dbo.LVCSRTrainingMaterial WITH (HOLDLOCK
TABLOCKX)')
GO
DROP TABLE dbo.LVCSRTrainingMaterial
GO
EXECUTE sp_rename N'dbo.Tmp_LVCSRTrainingMaterial',
N'LVCSRTrainingMaterial', 'OBJECT'
GO
COMMIT

          GO

update RecognitionLanguages set Display = 'English - USA' where
index1 = 0
update RecognitionLanguages set Display = 'Spanish - USA' where
index1 = 1
update RecognitionLanguages set Display = 'Spanish' where index1 = 2
update RecognitionLanguages set Display = 'English - British' where
index1 = 3
update RecognitionLanguages set Display = 'French - Canadian' where
index1 = 4
update RecognitionLanguages set Display = 'French' where index1 = 5
update RecognitionLanguages set Display = 'German' where index1 = 6
update RecognitionLanguages set Display = 'Italian' where index1 = 7
update RecognitionLanguages set Display = 'Portuguese - Brazil'
```

```
where index1 = 11
update RecognitionLanguages set Display = 'Catalan' where index1 =
13
update RecognitionLanguages set Display = 'Korean' where index1 = 15
update RecognitionLanguages set Display = 'English – South African'
where index1 = 16
update RecognitionLanguages set Display = 'Japanese' where index1 =
17
Go
          GO

update RecognitionLanguages set DictionaryName = null where index1
= 17
Go
          GO


ALTER TABLE [dbo].[TextStatus] DROP CONSTRAINT
[FK_TextStatus_TextData]
GO

ALTER TABLE [dbo].[TextData] DROP CONSTRAINT [PK_TextDataTbl]
GO

ALTER TABLE [dbo].[TextData] ADD  CONSTRAINT [PK_TextDataTbl]
PRIMARY KEY NONCLUSTERED
(
        [textId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, ONLINE = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
GO



ALTER TABLE [dbo].[TextStatus]  WITH NOCHECK ADD  CONSTRAINT
[FK_TextStatus_TextData] FOREIGN KEY([textId])
REFERENCES [dbo].[TextData] ([textId])
GO

ALTER TABLE [dbo].[TextStatus] NOCHECK CONSTRAINT
[FK_TextStatus_TextData]
```

```
GO

CREATE CLUSTERED INDEX [IX_originalTime] ON [dbo].[TextData]
(
        [originalTime] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, DROP_EXISTING = OFF,
ONLINE = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON
[PRIMARY]

GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- this SP creates the schema and the function for DB partitioning,
and set callMetaTbl, textMeta and repCategoryMetaTbl to use them
ALTER PROCEDURE [dbo].[sp_create_DB_storage_partitions]
AS
BEGIN
        -- SET NOCOUNT ON added to prevent extra result sets from
        -- interfering with SELECT statements.
        SET NOCOUNT ON;
        declare @statement as nvarchar(max)
        set @statement = '
        CREATE PARTITION FUNCTION fnPartitionCalls (int)
        AS RANGE RIGHT
        FOR VALUES (
        dbo.time2tod(''1/1/2014''),
        dbo.time2tod(''1/11/2014''),
        dbo.time2tod(''1/21/2014''),
        dbo.time2tod(''2/1/2014''),
        dbo.time2tod(''2/11/2014''),
        dbo.time2tod(''2/21/2014''),
        dbo.time2tod(''3/1/2014''),
        dbo.time2tod(''3/11/2014''),
        dbo.time2tod(''3/21/2014''),
        dbo.time2tod(''4/1/2014''),
        dbo.time2tod(''4/11/2014''),
        dbo.time2tod(''4/21/2014''),
        dbo.time2tod(''5/1/2014''),
        dbo.time2tod(''5/11/2014''),
```

```
dbo.time2tod(''5/21/2014''),
dbo.time2tod(''6/1/2014''),
dbo.time2tod(''6/11/2014''),
dbo.time2tod(''6/21/2014''),
dbo.time2tod(''7/1/2014''),
dbo.time2tod(''7/11/2014''),
dbo.time2tod(''7/21/2014''),
dbo.time2tod(''8/1/2014''),
dbo.time2tod(''8/11/2014''),
dbo.time2tod(''8/21/2014''),
dbo.time2tod(''9/1/2014''),
dbo.time2tod(''9/11/2014''),
dbo.time2tod(''9/21/2014''),
dbo.time2tod(''10/1/2014''),
dbo.time2tod(''10/11/2014''),
dbo.time2tod(''10/21/2014''),
dbo.time2tod(''11/1/2014''),
dbo.time2tod(''11/11/2014''),
dbo.time2tod(''11/21/2014''),
dbo.time2tod(''12/1/2014''),
dbo.time2tod(''12/11/2014''),
dbo.time2tod(''12/21/2014''),
dbo.time2tod(''1/1/2015''),
dbo.time2tod(''1/11/2015''),
dbo.time2tod(''1/21/2015''),
dbo.time2tod(''2/1/2015''),
dbo.time2tod(''2/11/2015''),
dbo.time2tod(''2/21/2015''),
dbo.time2tod(''3/1/2015''),
dbo.time2tod(''3/11/2015''),
dbo.time2tod(''3/21/2015''),
dbo.time2tod(''4/1/2015''),
dbo.time2tod(''4/11/2015''),
dbo.time2tod(''4/21/2015''),
dbo.time2tod(''5/1/2015''),
dbo.time2tod(''5/11/2015''),
dbo.time2tod(''5/21/2015''),
dbo.time2tod(''6/1/2015''),
dbo.time2tod(''6/11/2015''),
dbo.time2tod(''6/21/2015''),
dbo.time2tod(''7/1/2015''),
dbo.time2tod(''7/11/2015''),
```

```
dbo.time2tod(''7/21/2015''),
dbo.time2tod(''8/1/2015''),
dbo.time2tod(''8/11/2015''),
dbo.time2tod(''8/21/2015''),
dbo.time2tod(''9/1/2015''),
dbo.time2tod(''9/11/2015''),
dbo.time2tod(''9/21/2015''),
dbo.time2tod(''10/1/2015''),
dbo.time2tod(''10/11/2015''),
dbo.time2tod(''10/21/2015''),
dbo.time2tod(''11/1/2015''),
dbo.time2tod(''11/11/2015''),
dbo.time2tod(''11/21/2015''),
dbo.time2tod(''12/1/2015''),
dbo.time2tod(''12/11/2015''),
dbo.time2tod(''12/21/2015''),
dbo.time2tod(''1/1/2016''),
dbo.time2tod(''1/11/2016''),
dbo.time2tod(''1/21/2016''),
dbo.time2tod(''2/1/2016''),
dbo.time2tod(''2/11/2016''),
dbo.time2tod(''2/21/2016''),
dbo.time2tod(''3/1/2016''),
dbo.time2tod(''3/11/2016''),
dbo.time2tod(''3/21/2016''),
dbo.time2tod(''4/1/2016''),
dbo.time2tod(''4/11/2016''),
dbo.time2tod(''4/21/2016''),
dbo.time2tod(''5/1/2016''),
dbo.time2tod(''5/11/2016''),
dbo.time2tod(''5/21/2016''),
dbo.time2tod(''6/1/2016''),
dbo.time2tod(''6/11/2016''),
dbo.time2tod(''6/21/2016''),
dbo.time2tod(''7/1/2016''),
dbo.time2tod(''7/11/2016''),
dbo.time2tod(''7/21/2016''),
dbo.time2tod(''8/1/2016''),
dbo.time2tod(''8/11/2016''),
dbo.time2tod(''8/21/2016''),
dbo.time2tod(''9/1/2016''),
dbo.time2tod(''9/11/2016''),
```

```
dbo.time2tod(''9/21/2016''),
dbo.time2tod(''10/1/2016''),
dbo.time2tod(''10/11/2016''),
dbo.time2tod(''10/21/2016''),
dbo.time2tod(''11/1/2016''),
dbo.time2tod(''11/11/2016''),
dbo.time2tod(''11/21/2016''),
dbo.time2tod(''12/1/2016''),
dbo.time2tod(''12/11/2016''),
dbo.time2tod(''12/21/2016''),
dbo.time2tod(''1/1/2017''),
dbo.time2tod(''1/11/2017''),
dbo.time2tod(''1/21/2017''),
dbo.time2tod(''2/1/2017''),
dbo.time2tod(''2/11/2017''),
dbo.time2tod(''2/21/2017''),
dbo.time2tod(''3/1/2017''),
dbo.time2tod(''3/11/2017''),
dbo.time2tod(''3/21/2017''),
dbo.time2tod(''4/1/2017''),
dbo.time2tod(''4/11/2017''),
dbo.time2tod(''4/21/2017''),
dbo.time2tod(''5/1/2017''),
dbo.time2tod(''5/11/2017''),
dbo.time2tod(''5/21/2017''),
dbo.time2tod(''6/1/2017''),
dbo.time2tod(''6/11/2017''),
dbo.time2tod(''6/21/2017''),
dbo.time2tod(''7/1/2017''),
dbo.time2tod(''7/11/2017''),
dbo.time2tod(''7/21/2017''),
dbo.time2tod(''8/1/2017''),
dbo.time2tod(''8/11/2017''),
dbo.time2tod(''8/21/2017''),
dbo.time2tod(''9/1/2017''),
dbo.time2tod(''9/11/2017''),
dbo.time2tod(''9/21/2017''),
dbo.time2tod(''10/1/2017''),
dbo.time2tod(''10/11/2017''),
dbo.time2tod(''10/21/2017''),
dbo.time2tod(''11/1/2017''),
dbo.time2tod(''11/11/2017''),
```

```
dbo.time2tod(''11/21/2017''),
dbo.time2tod(''12/1/2017''),
dbo.time2tod(''12/11/2017''),
dbo.time2tod(''12/21/2017''),
dbo.time2tod(''1/1/2018''),
dbo.time2tod(''1/11/2018''),
dbo.time2tod(''1/21/2018''),
dbo.time2tod(''2/1/2018''),
dbo.time2tod(''2/11/2018''),
dbo.time2tod(''2/21/2018''),
dbo.time2tod(''3/1/2018''),
dbo.time2tod(''3/11/2018''),
dbo.time2tod(''3/21/2018''),
dbo.time2tod(''4/1/2018''),
dbo.time2tod(''4/11/2018''),
dbo.time2tod(''4/21/2018''),
dbo.time2tod(''5/1/2018''),
dbo.time2tod(''5/11/2018''),
dbo.time2tod(''5/21/2018''),
dbo.time2tod(''6/1/2018''),
dbo.time2tod(''6/11/2018''),
dbo.time2tod(''6/21/2018''),
dbo.time2tod(''7/1/2018''),
dbo.time2tod(''7/11/2018''),
dbo.time2tod(''7/21/2018''),
dbo.time2tod(''8/1/2018''),
dbo.time2tod(''8/11/2018''),
dbo.time2tod(''8/21/2018''),
dbo.time2tod(''9/1/2018''),
dbo.time2tod(''9/11/2018''),
dbo.time2tod(''9/21/2018''),
dbo.time2tod(''10/1/2018''),
dbo.time2tod(''10/11/2018''),
dbo.time2tod(''10/21/2018''),
dbo.time2tod(''11/1/2018''),
dbo.time2tod(''11/11/2018''),
dbo.time2tod(''11/21/2018''),
dbo.time2tod(''12/1/2018''),
dbo.time2tod(''12/11/2018''),
dbo.time2tod(''12/21/2018''),
dbo.time2tod(''12/21/2028'')
)
```

```
        CREATE PARTITION SCHEME CallScheme
        AS PARTITION fnPartitionCalls
        ALL to ([Primary])

        DROP INDEX [IX_callTime] ON [dbo].[callMetaTbl]

        CREATE CLUSTERED INDEX [IX_callTime] ON [dbo].[callMetaTbl]
        (
                [callTime] ASC
        )WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, DROP_EXISTING = OFF,
ONLINE = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON
[CallScheme]([callTime])


        DROP INDEX [IX_callTime] ON [dbo].[repCategoryMetaTbl]

        CREATE CLUSTERED INDEX [IX_callTime] ON
[dbo].[repCategoryMetaTbl]
        (
                [callTime] ASC
        )WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, DROP_EXISTING = OFF,
ONLINE = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON
[CallScheme]([callTime])

        DROP INDEX [IX_originalTime] ON [dbo].[TextData]

        CREATE CLUSTERED INDEX [IX_originalTime] ON
[dbo].[TextData]
        (
                [originalTime] ASC
        )WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, DROP_EXISTING = OFF,
ONLINE = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON
[CallScheme]([originalTime])'

        IF CHARINDEX('Enterprise', CONVERT(varchar(1000),
SERVERPROPERTY('edition'))) > 0
        BEGIN
                EXEC dbo.sp_executesql @statement
```

```
        End
END
GO

          GO

update audioConversionTypesTbl set
conversionType=4,externalToolPath=null,parametersForTool=null where
fromFormat=3 and toFormat=3
          GO

/****** Object:  StoredProcedure [dbo].[sp_createMaintenanceJob]
   Script Date: 9/17/2014 11:40:40 AM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

ALTER PROCEDURE [dbo].[sp_createMaintenanceJob]
  @targetSpeechMinerDB varchar(50)
AS
BEGIN

DECLARE @connectorService varchar(256)
DECLARE @connectorComputer varchar(24)

-------------------------------------------------------------------------
--------- Set the connector service name and computer for the job
to stop the connector before continuing maintenance ---------
-------------------------------------------------------------------------
-------------------------------------------------------------------------
--------- Turn this flag on to create the agents tree from the
calls, if you do not have the agent information ---------
-------------------------------------------------------------------------

DECLARE @createAgentsFromPartitions bit
SET @createAgentsFromPartitions = 0
DECLARE @createAgentsDaysToKeep int
SET @createAgentsDaysToKeep = 30

/****** Object:  Job [SpeechMiner_Maintenance]    ******/
```

```
BEGIN TRANSACTION
DECLARE @jobName VARCHAR(100)
SET @jobName='SpeechMiner_Maintenance_job - ' + @targetSpeechMinerDB
DECLARE @oldJobId AS uniqueidentifier
SELECT @oldJobId = job_id FROM msdb.dbo.sysjobs_view WHERE name =
@jobName
IF @oldJobId IS NOT NULL
EXEC msdb.dbo.sp_delete_job @job_id=@oldJobId,
@delete_unused_schedule=1
DECLARE @ReturnCode INT
SELECT @ReturnCode = 0
/****** Object:  JobCategory [Database Maintenance]    ******/
IF NOT EXISTS (SELECT name FROM msdb.dbo.syscategories WHERE
name=N'Database Maintenance' AND category_class=1)
BEGIN
EXEC @ReturnCode = msdb.dbo.sp_add_category @class=N'JOB',
@type=N'LOCAL', @name=N'Database Maintenance'
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
END

DECLARE @jobId BINARY(16)
EXEC @ReturnCode =  msdb.dbo.sp_add_job @job_name=@jobName,
                @enabled=1,
                @notify_level_eventlog=0,
                @notify_level_email=0,
                @notify_level_netsend=0,
                @notify_level_page=0,
                @delete_level=0,
                @description=N'No description available.',
                @category_name=N'Database Maintenance',
                @owner_login_name=N'dbuser', @job_id = @jobId OUTPUT

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Shutdown System]    ******/
DECLARE @stopCmd as nvarchar(max)
SET @stopCmd = 'update computerlist set status = 0'
IF @connectorService is not null AND @connectorComputer is not null
BEGIN
  SET @stopCmd = @stopCmd + '
  declare @r as int
  exec @r =  dbo.sp_control_service 0,''' + @connectorService +
```

```
''',''' + @connectorComputer + '''
  select ''result of stopping connector ''+ Cast(@r as nvarchar(1))'
END

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Shutdown System',
                @step_id=1,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=4,
                @on_fail_step_id=4,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=@stopCmd,
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Rebuild Index]    ******/
EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Rebuild Index',
                @step_id=2,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=3,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=N'EXECUTE sp_rebuild_indexes',
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Turn on System]    ******/

DECLARE @startCmd as nvarchar(max)
```

```
SET @startCmd = 'update computerlist set status = 1'
IF @connectorService is not null AND @connectorComputer is not null
BEGIN
  SET @startCmd = @startCmd + '
  declare @r as int
  exec @r =  dbo.sp_control_service 1,''' + @connectorService +
''',''' + @connectorComputer + '''

  select ''result of start uconnector ''+ Cast(@r as nvarchar(1))'
END

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Turn on System',
                @step_id=3,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=3,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=@startCmd,
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Purge old msg logs]    ******/

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Purge old msg logs',
                @step_id=4,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=3,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
```

```
                @command=N'delete from msgLogTbl
                where [time] < dateadd(month,-1,getdate())',
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Purge old user events]    ******/

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Purge old user events',
                @step_id=5,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=3,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=N'delete from userEventsTbl
                where eventTime < dateadd(month,-12,getdate())',
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/****** Object:  Step [Purge report agents filter]    ******/

DECLARE @agentsCmd as nvarchar(max)

SET @agentsCmd = ''

IF @createAgentsFromPartitions = 1
BEGIN
  SET @agentsCmd = @agentsCmd + '
  exec sp_createAgentsFromPartitions ' +
cast(@createAgentsDaysToKeep as varchar(10))
END

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Update agents',
```

```
                @step_id=6,
                @cmdexec_success_code=0,
                @on_success_action=3,
                @on_success_step_id=0,
                @on_fail_action=3,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=@agentsCmd,
                @database_name=@targetSpeechMinerDB,
                @flags=0


IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback


/****** Object:  Step [Clean stuck calls]    ******/

DECLARE @CleanCallsCmd as nvarchar(max)

SET @CleanCallsCmd = 'update CallStatusTbl set startrectime=0
where startrectime=-1 and endRecTime=0
and not exists (select callid from CallQTbl where
CallQTbl.callid=CallStatusTbl.callid)'

EXEC @ReturnCode = msdb.dbo.sp_add_jobstep @job_id=@jobId,
@step_name=N'Clean stuck calls',
                @step_id=7,
                @cmdexec_success_code=0,
                @on_success_action=1,
                @on_success_step_id=0,
                @on_fail_action=1,
                @on_fail_step_id=0,
                @retry_attempts=0,
                @retry_interval=0,
                @os_run_priority=0, @subsystem=N'TSQL',
                @command=@CleanCallsCmd,
                @database_name=@targetSpeechMinerDB,
                @flags=0

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
```

```
/*********************************************/

EXEC @ReturnCode = msdb.dbo.sp_update_job @job_id = @jobId,
@start_step_id = 1

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

/* Uncomment to schedule job (parameters are for weekly run)

EXEC @ReturnCode = msdb.dbo.sp_add_jobschedule @job_id=@jobId,
@name=N'Weekly',
                @enabled=1,
                @freq_type=8,
                @freq_interval=1,
                @freq_subday_type=1,
                @freq_subday_interval=0,
                @freq_relative_interval=0,
                @freq_recurrence_factor=1,
                @active_start_date=20070820,
                @active_end_date=99991231,
                @active_start_time=0,
                @active_end_time=235959

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

*/

EXEC @ReturnCode = msdb.dbo.sp_add_jobserver @job_id = @jobId,
@server_name = N'(local)'

IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback

COMMIT TRANSACTION
GOTO EndSave

QuitWithRollback:

    IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION

EndSave:
```

```
END
          GO

grant select on ComputerList to CreatePerfCounters
          GO

ALTER TABLE wordPronunciation
ALTER COLUMN Word nvarchar(64)

Insert into wordPronunciation
values(17,N'????','PREDEFINED','bIbIta4M')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','yamayama')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','masMmasM')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','wa4Ewa4E')
Insert into wordPronunciation values(17,N'??','PREDEFINED','I4OI4O')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tabItabI')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','samazama')
Insert into wordPronunciation values(17,N'??','PREDEFINED','mENmEN')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','yOyOgI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','atOatO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','kOmagOma')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','dZMMdZMM')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','ENENtO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tsMnEzMnE')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','mOkMmOkMtO')
Insert into wordPronunciation values(17,N'??','PREDEFINED','SOOSOO')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','tSIkadZIkanO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','mO4OmO4O')
```

```
Insert into wordPronunciation values(17,N'??','PREDEFINED','OOOO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','ItSIItSI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','kIgI')
Insert into wordPronunciation
values(17,N'????','PREDEFINED','namanamaSII')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','naganaga')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','ma4Mma4M')
Insert into wordPronunciation values(17,N'??','PREDEFINED','MNnMN')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','sakIzakI')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','sasakI')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tsMgItsMgI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','sOOsOO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','mOtOmOtO')
Insert into wordPronunciation values(17,N'??','PREDEFINED','saNzaN')
Insert into wordPronunciation values(17,N'???','PREDEFINED','SIsMI')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tOkIdOkI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','daNdaN')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','kazMkazM')
Insert into wordPronunciation values(17,N'??','PREDEFINED','hIbI')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','zOkMzOkM')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','kMnIgMnI')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','hIsabIsa')
Insert into wordPronunciation values(17,N'??','PREDEFINED','taNtaN')
Insert into wordPronunciation values(17,N'??','PREDEFINED','dOOdOO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tSIkadZIka')
Insert into wordPronunciation values(17,N'??','PREDEFINED','tOOtOO')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tsMkIzMkI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','SMdZM')
```

```
Insert into wordPronunciation
values(17,N'??','PREDEFINED','tSakMtSakM')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','katagata')
Insert into wordPronunciation values(17,N'??','PREDEFINED','nENnEN')
Insert into wordPronunciation
values(17,N'??','PREDEFINED','hItObItO')
Insert into wordPronunciation values(17,N'??','PREDEFINED','kOkO')
Insert into wordPronunciation values(17,N'??','PREDEFINED','tata')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','dZOdZOnI')
Insert into wordPronunciation values(17,N'??','PREDEFINED','tENtEN')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','daIdaItEkI')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','kOkOdZIN')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','I4OI4OO')
Insert into wordPronunciation
values(17,N'???','PREDEFINED','tOkIdOkII')

          GO

GRANT select ON [dbo].[wordPronunciation] TO
[SMARTCompileGrammar],[InteractionReceiver],[Platform],[SMConfig],[Web]

GO

          GO

/****** Adding filter words in Korean ******/

  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  insert into wordFilterTbl values (17, N'?????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
```

```
insert into wordFilterTbl values (17, N'????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'???', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'???', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'??????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'??????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'???', 0)
insert into wordFilterTbl values (17, N'?', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
insert into wordFilterTbl values (17, N'?????', 0)
insert into wordFilterTbl values (17, N'??', 0)
```

```
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  insert into wordFilterTbl values (17, N'????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'????', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  insert into wordFilterTbl values (17, N'??', 0)
  insert into wordFilterTbl values (17, N'?', 0)
  GO
         GO

/****** Object:  Table [dbo].[alertMailMessageTbl]    Script Date:
09/22/2014 11:01:48 ******/

/****** Object:  Table [dbo].[alertMailMessageTbl]    Script Date:
09/22/2014 11:01:48 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_alertMailMessageTbl](
        [ID] [int] NOT NULL,
        [Text] [nvarchar](256) NOT NULL
) ON [PRIMARY]

GO

SET ANSI_PADDING ON
GO

GRANT SELECT ON dbo.[tmp_alertMailMessageTbl] TO Web,utopy  AS dbo
GO
GRANT INSERT ON dbo.[tmp_alertMailMessageTbl] TO utopy  AS dbo
```

```
GO
GRANT DELETE ON dbo.[tmp_alertMailMessageTbl] TO utopy  AS dbo
GO
GRANT UPDATE ON dbo.[tmp_alertMailMessageTbl] TO utopy  AS dbo
GO

IF EXISTS(SELECT * FROM dbo.alertMailMessageTbl)
         EXEC('INSERT INTO dbo.[tmp_alertMailMessageTbl] (ID,[Text])
                 SELECT ID, CONVERT(nvarchar(256), Text) FROM
dbo.alertMailMessageTbl WITH (HOLDLOCK TABLOCKX)')
GO


IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[alertMailMessageTbl]') AND type in (N'U'))
DROP TABLE [dbo].[alertMailMessageTbl]
GO

EXECUTE sp_rename N'dbo.tmp_alertMailMessageTbl',
N'alertMailMessageTbl', 'OBJECT'


 ALTER TABLE alertMailMessageTbl ADD CONSTRAINT
[PK_alertMailMessageTbl] PRIMARY KEY CLUSTERED
(
        [ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_alertTopicTbl_alertTbl]') AND
parent_object_id = OBJECT_ID(N'[dbo].[alertTopicTbl]'))
ALTER TABLE [dbo].[alertTopicTbl] DROP CONSTRAINT
[FK_alertTopicTbl_alertTbl]
GO
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_alertCategoryTbl_alertTbl]') AND
parent_object_id = OBJECT_ID(N'[dbo].[alertCategoryTbl]'))
ALTER TABLE [dbo].[alertCategoryTbl] DROP CONSTRAINT
```

```
[FK_alertCategoryTbl_alertTbl]
GO
IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF__alertTbl__timeCr__4D4C0586]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[alertTbl] DROP CONSTRAINT
[DF__alertTbl__timeCr__4D4C0586]
END

GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF__alertTbl__timeUp__4E4029BF]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[alertTbl] DROP CONSTRAINT
[DF__alertTbl__timeUp__4E4029BF]
END


/****** Object:  Table [dbo].[alertTbl]    Script Date: 09/22/2014
12:04:37 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_alertTbl](
        [alertId] [int] IDENTITY(1000,1) NOT NULL,
        [alertName] [nvarchar](256) NULL,
        [creator] [nvarchar](256) NOT NULL,
        [timeCreated] [datetime] NOT NULL,
        [timeUpdated] [datetime] NOT NULL,
        [lastMonitorTime] [datetime] NULL,
        [lastAlertTime] [datetime] NULL,
        [alertTypeId] [int] NOT NULL,
        [alertType] [int] NOT NULL) ON [PRIMARY]

GO
```

```
GO
GRANT SELECT ON dbo.[tmp_alertTbl] TO Platform,utopy  AS dbo
GO
GRANT INSERT ON dbo.[tmp_alertTbl] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_alertTbl] TO utopy  AS dbo
GO
GRANT UPDATE ON dbo.[tmp_alertTbl] TO utopy  AS dbo
GO
GRANT UPDATE ON dbo.[tmp_alertTbl](lastMonitorTime) TO platform  AS
dbo
GO

GRANT VIEW DEFINITION ON dbo.[tmp_alertTbl] TO utopy  AS dbo
GO
SET ANSI_PADDING ON
GO
SET IDENTITY_INSERT dbo.[tmp_alertTbl] ON

IF EXISTS(SELECT * FROM [dbo].alertTbl)
         EXEC('INSERT INTO dbo.[tmp_alertTbl]
(alertId,[alertName],[creator],[timeCreated],[timeUpdated],[lastMonitorTime],
                SELECT alertId, CONVERT(nvarchar(256), alertName),
CONVERT(nvarchar(1000),
creator),[timeCreated],[timeUpdated],[lastMonitorTime],alertTypeId,[alertType
FROM dbo.[alertTbl] WITH (HOLDLOCK TABLOCKX)')
GO

SET IDENTITY_INSERT dbo.[tmp_alertTbl] OFF
GO
/****** Object:  Table [dbo].[alertTbl]    Script Date: 09/22/2014
12:04:37 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[alertTbl]') AND type in (N'U'))
DROP TABLE [dbo].[alertTbl]
GO

EXECUTE sp_rename N'dbo.tmp_alertTbl', N'alertTbl', 'OBJECT'
GO
ALTER TABLE [dbo].[alertTbl] ADD CONSTRAINT [PK_alertTbl] PRIMARY
KEY CLUSTERED
(
```

```
        [alertId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]

ALTER TABLE [dbo].[alertTbl] ADD  CONSTRAINT
[DF__alertTbl__timeCr__4D4C0586]  DEFAULT (getutcdate()) FOR
[timeCreated]
GO

ALTER TABLE [dbo].[alertTbl] ADD  CONSTRAINT
[DF__alertTbl__timeUp__4E4029BF]  DEFAULT (getutcdate()) FOR
[timeUpdated]
GO


ALTER TABLE [dbo].[alertTopicTbl]  WITH CHECK ADD  CONSTRAINT
[FK_alertTopicTbl_alertTbl] FOREIGN KEY([alertId])
REFERENCES [dbo].[alertTbl] ([alertId])
GO

ALTER TABLE [dbo].[alertTopicTbl] CHECK CONSTRAINT
[FK_alertTopicTbl_alertTbl]
GO
ALTER TABLE [dbo].[alertCategoryTbl]  WITH CHECK ADD  CONSTRAINT
[FK_alertCategoryTbl_alertTbl] FOREIGN KEY([alertId])
REFERENCES [dbo].[alertTbl] ([alertId])
GO

ALTER TABLE [dbo].[alertCategoryTbl] CHECK CONSTRAINT
[FK_alertCategoryTbl_alertTbl]
GO

EXEC sys.sp_addextendedproperty @name=N'MS_Description',
@value=N'Link between alertTbl and alertCategoryTbl' ,
@level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'TABLE',@level1name=N'alertCategoryTbl',
@level2type=N'CONSTRAINT',@level2name=N'FK_alertCategoryTbl_alertTbl'
GO
          GO
```

```
/****** Object:  Table [dbo].[alertUsersTbl]    Script Date: 09/22/
2014 12:01:25 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_alertUsersTbl](
        [userLogin] [nvarchar](256) NOT NULL,
        [alertId] [int] NOT NULL) ON [PRIMARY]

GO

SET ANSI_PADDING ON
GO


GRANT SELECT ON dbo.[tmp_alertUsersTbl] TO Platform,utopy  AS dbo
GO
GRANT INSERT ON dbo.[tmp_alertUsersTbl] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_alertUsersTbl] TO utopy  AS dbo
GO


IF EXISTS(SELECT * FROM dbo.alertUsersTbl)
         EXEC('INSERT INTO dbo.[tmp_alertUsersTbl] (ID,[Text])
                 SELECT ID, CONVERT(nvarchar(256), Text) FROM
dbo.alertUsersTbl WITH (HOLDLOCK TABLOCKX)')
GO


/****** Object:  Table [dbo].[alertUsersTbl]    Script Date: 09/22/
2014 12:01:25 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[alertUsersTbl]') AND type in (N'U'))
```

```
DROP TABLE [dbo].[alertUsersTbl]
GO

EXECUTE sp_rename N'dbo.tmp_alertUsersTbl', N'alertUsersTbl',
'OBJECT'


ALTER TABLE alertUsersTbl ADD CONSTRAINT
[PK__alertUsersTbl__5AA469F6] PRIMARY KEY CLUSTERED
(
        [userLogin] ASC,
        [alertId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
          GO




/****** Object:  Table [dbo].[alertUsersTbl]    Script Date: 09/22/
2014 12:01:25 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_alertUsersTbl](
        [userLogin] [nvarchar](256) NOT NULL,
        [alertId] [int] NOT NULL) ON [PRIMARY]

GO

SET ANSI_PADDING ON
GO


GRANT SELECT ON dbo.[tmp_alertUsersTbl] TO Platform,utopy  AS dbo
```

```
GO
GRANT INSERT ON dbo.[tmp_alertUsersTbl] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_alertUsersTbl] TO utopy  AS dbo
GO


IF EXISTS(SELECT * FROM dbo.alertUsersTbl)
         EXEC('INSERT INTO dbo.[tmp_alertUsersTbl] (ID,[Text])
                 SELECT ID, CONVERT(nvarchar(256), Text) FROM
dbo.alertUsersTbl WITH (HOLDLOCK TABLOCKX)')
GO


/****** Object:  Table [dbo].[alertUsersTbl]    Script Date: 09/22/
2014 12:01:25 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[alertUsersTbl]') AND type in (N'U'))
DROP TABLE [dbo].[alertUsersTbl]
GO

EXECUTE sp_rename N'dbo.tmp_alertUsersTbl', N'alertUsersTbl',
'OBJECT'


ALTER TABLE alertUsersTbl ADD CONSTRAINT
[PK__alertUsersTbl__5AA469F6] PRIMARY KEY CLUSTERED
(
        [userLogin] ASC,
        [alertId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingSessionsTbl_coachingSessionNotesTbl]')
AND parent_object_id = OBJECT_ID(N'[dbo].[coachingSessionNotes]'))
ALTER TABLE [dbo].[coachingSessionNotes] DROP CONSTRAINT
[FK_coachingSessionsTbl_coachingSessionNotesTbl]
GO
```

```
IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingSessionNotes_timeCreated]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessionNotes] DROP CONSTRAINT
[DF_coachingSessionNotes_timeCreated]
END

GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingSessionNotes_timeUpdated]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessionNotes] DROP CONSTRAINT
[DF_coachingSessionNotes_timeUpdated]
END

GO
/****** Object:  Table [dbo].[coachingSessionNotes]    Script Date:
09/18/2014 11:05:14 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_coachingSessionNotes](
        [ID] [int] IDENTITY(1000,1) NOT NULL,
        [creator] [nvarchar](256) NOT NULL,
        [timeCreated] [datetime] NOT NULL,
        [timeUpdated] [datetime] NOT NULL,
        [noteText] [nvarchar](max) NOT NULL,
        [sessionId] [int] NOT NULL,
        [permissions] [nvarchar](100) NOT NULL) ON [PRIMARY]
        GO
ALTER TABLE dbo.[tmp_coachingSessionNotes] SET (LOCK_ESCALATION =
TABLE)
GO
SET IDENTITY_INSERT dbo.[tmp_coachingSessionNotes] ON
```

```
GO
GRANT SELECT ON dbo.[tmp_coachingSessionNotes] TO reports,utopy  AS
dbo
GO
GRANT INSERT ON dbo.[tmp_coachingSessionNotes] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_coachingSessionNotes] TO utopy,web  AS dbo
GO
GRANT UPDATE ON dbo.[tmp_coachingSessionNotes] TO utopy  AS dbo
go
GRANT UPDATE ON dbo.[tmp_coachingSessionNotes]  TO web  AS dbo
GO
IF EXISTS(SELECT * FROM dbo.coachingSessionNotes)
         EXEC('INSERT INTO dbo.[tmp_coachingSessionNotes]
(Id,creator,timeCreated,timeUpdated,noteText,sessionId,permissions)
                SELECT Id,CONVERT(nvarchar(256),
creator),timeCreated,timeUpdated, CONVERT(nvarchar(max), noteText),
sessionId, CONVERT(nvarchar(100), permissions) FROM
dbo.[coachingSessionNotes] WITH (HOLDLOCK TABLOCKX)')
GO

SET IDENTITY_INSERT dbo.[tmp_coachingSessionNotes] OFF
GO
SET ANSI_PADDING ON
GO
/****** Object:  Table [dbo].[coachingSessionNotes]    Script Date:
09/18/2014 11:05:14 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[coachingSessionNotes]') AND type in (N'U'))
DROP TABLE [dbo].[coachingSessionNotes]
GO
EXECUTE sp_rename N'dbo.tmp_coachingSessionNotes',
N'coachingSessionNotes', 'OBJECT'
go

ALTER TABLE [dbo].[coachingSessionNotes] ADD CONSTRAINT
[PK_coachingSessionNotesTbl] PRIMARY KEY CLUSTERED
(
        [ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
```

```
ALTER TABLE [dbo].[coachingSessionNotes]  WITH CHECK ADD
 CONSTRAINT [FK_coachingSessionsTbl_coachingSessionNotesTbl]
FOREIGN KEY([sessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingSessionNotes] CHECK CONSTRAINT
[FK_coachingSessionsTbl_coachingSessionNotesTbl]
GO

ALTER TABLE [dbo].[coachingSessionNotes] ADD  CONSTRAINT
[DF_coachingSessionNotes_timeCreated]  DEFAULT (getutcdate()) FOR
[timeCreated]
GO

ALTER TABLE [dbo].[coachingSessionNotes] ADD  CONSTRAINT
[DF_coachingSessionNotes_timeUpdated]  DEFAULT (getutcdate()) FOR
[timeUpdated]
GO


          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_CoachingSessionTypeTbl_coachingSessions]')
AND parent_object_id = OBJECT_ID(N'[dbo].[coachingSessions]'))
ALTER TABLE [dbo].[coachingSessions] DROP CONSTRAINT
[FK_CoachingSessionTypeTbl_coachingSessions]
GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingSessions_timeCreated]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessions] DROP CONSTRAINT
[DF_coachingSessions_timeCreated]
END

GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
```

```
OBJECT_ID(N'[DF_coachingSessions_timeUpdated]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessions] DROP CONSTRAINT
[DF_coachingSessions_timeUpdated]
END

GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingSessions_accepted]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessions] DROP CONSTRAINT
[DF_coachingSessions_accepted]
END

GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF__coachingS__isAct__642DD430]') AND type = 'D')
BEGIN
ALTER TABLE [dbo].[coachingSessions] DROP CONSTRAINT
[DF__coachingS__isAct__642DD430]
END
GO
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingDynamicCallLists_coachingDynamicCallLists]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingDynamicCallLists]'))
ALTER TABLE [dbo].[coachingDynamicCallLists] DROP CONSTRAINT
[FK_coachingDynamicCallLists_coachingDynamicCallLists]
GO
GO

IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingSessionActionItems_coachingSessions]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingSessionActionItems]'))
ALTER TABLE [dbo].[coachingSessionActionItems] DROP CONSTRAINT
[FK_coachingSessionActionItems_coachingSessions]
GO
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingSessionsTbl_coachingSessionNotesTbl]')
```

```
AND parent_object_id = OBJECT_ID(N'[dbo].[coachingSessionNotes]'))
ALTER TABLE [dbo].[coachingSessionNotes] DROP CONSTRAINT
[FK_coachingSessionsTbl_coachingSessionNotesTbl]
GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingSessionResources_coachingSessions]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingSessionResources]'))
ALTER TABLE [dbo].[coachingSessionResources] DROP CONSTRAINT
[FK_coachingSessionResources_coachingSessions]
GO
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingStaticCallLists_coachingSessions]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingStaticCallLists]'))
ALTER TABLE [dbo].[coachingStaticCallLists] DROP CONSTRAINT
[FK_coachingStaticCallLists_coachingSessions]
GO
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_FormsAnsweredTbl_coachingSessions]') AND
parent_object_id = OBJECT_ID(N'[dbo].[FormsAnsweredTbl]'))
ALTER TABLE [dbo].[FormsAnsweredTbl] DROP CONSTRAINT
[FK_FormsAnsweredTbl_coachingSessions]
GO
IF  EXISTS (SELECT * FROM sys.views WHERE object_id =
OBJECT_ID(N'[dbo].[userMessagesView]'))
DROP VIEW [dbo].[userMessagesView]
GO
CREATE TABLE [dbo].[tmp_coachingSessions](
        [sessionId] [int] IDENTITY(1000,1) NOT NULL,
        [name] [nvarchar](256) NULL,
        [creator] [nvarchar](256) NOT NULL,
        [coach] [nvarchar](256) NULL,
        [trainee] [nvarchar](256) NULL,
        [dueDate] [datetime] NULL,
        [dueEnd] [datetime] NULL,
        [status] [nvarchar](20) NOT NULL,
        [ScheduleStart] [datetime] NULL,
        [ScheduleEnd] [datetime] NULL,
        [duration] [float] NULL,
```

```
        [timeCreated] [datetime] NOT NULL,
        [timeUpdated] [datetime] NOT NULL,
        [accepted] [bit] NOT NULL,
        [sessionType] [int] NULL,
        [isActive] [bit] NOT NULL,
        [ScheduleLastStart] [datetime] NULL) ON [PRIMARY]



SET ANSI_PADDING ON
GO
ALTER TABLE dbo.tmp_coachingSessions SET (LOCK_ESCALATION = TABLE)
GO
GO
GRANT SELECT ON dbo.tmp_coachingSessions TO Web,reports,utopy  AS
dbo
GO
GRANT INSERT ON dbo.tmp_coachingSessions TO utopy  AS dbo
GO
GRANT DELETE ON dbo.tmp_coachingSessions TO utopy,web  AS dbo
GO
GRANT UPDATE ON dbo.tmp_coachingSessions TO utopy  AS dbo
GO
GRANT VIEW DEFINITION ON dbo.tmp_coachingSessions TO utopy  AS dbo
GO
SET IDENTITY_INSERT dbo.tmp_coachingSessions ON
GO
IF EXISTS(SELECT * FROM dbo.coachingSessions)
         EXEC('INSERT INTO dbo.tmp_coachingSessions
([sessionId],[name],[creator],[coach],[trainee],[dueDate],[dueEnd],[status],[
                SELECT sessionId, CONVERT(nvarchar(256),
name),CONVERT(nvarchar(256), creator),CONVERT(nvarchar(256),
coach),CONVERT(nvarchar(256),
trainee),dueDate,dueEnd,CONVERT(nvarchar(20),
status),ScheduleStart,ScheduleEnd,duration,timeCreated,timeUpdated,accepted,s
FROM dbo.coachingSessions WITH (HOLDLOCK TABLOCKX)')
GO
SET IDENTITY_INSERT dbo.tmp_coachingSessions OFF
GO
/****** Object:  Table [dbo].[coachingSessions]    Script Date: 09/
18/2014 07:16:34 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
```

```
OBJECT_ID(N'[dbo].[coachingSessions]') AND type in (N'U'))
DROP TABLE [dbo].[coachingSessions]
GO

EXECUTE sp_rename N'dbo.tmp_coachingSessions', N'coachingSessions',
'OBJECT'
GO
 ALTER TABLE [coachingSessions] ADD CONSTRAINT
[PK_coachingSessionsTbl] PRIMARY KEY CLUSTERED
(
        [sessionId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]


GO
ALTER TABLE [dbo].[coachingSessions]  WITH CHECK ADD  CONSTRAINT
[FK_CoachingSessionTypeTbl_coachingSessions] FOREIGN
KEY([sessionType])
REFERENCES [dbo].[CoachingSessionTypeTbl] ([TypeID])
GO

ALTER TABLE [dbo].[coachingSessions] CHECK CONSTRAINT
[FK_CoachingSessionTypeTbl_coachingSessions]
GO

ALTER TABLE [dbo].[coachingSessions] ADD  CONSTRAINT
[DF_coachingSessions_timeCreated]  DEFAULT (getutcdate()) FOR
[timeCreated]
GO

ALTER TABLE [dbo].[coachingSessions] ADD  CONSTRAINT
[DF_coachingSessions_timeUpdated]  DEFAULT (getutcdate()) FOR
[timeUpdated]
GO

ALTER TABLE [dbo].[coachingSessions] ADD  CONSTRAINT
[DF_coachingSessions_accepted]  DEFAULT ((0)) FOR [accepted]
GO

ALTER TABLE [dbo].[coachingSessions] ADD  CONSTRAINT
```

```
[DF__coachingS__isAct__642DD430]  DEFAULT ((1)) FOR [isActive]
GO


ALTER TABLE [dbo].[coachingDynamicCallLists]  WITH CHECK ADD
 CONSTRAINT [FK_coachingDynamicCallLists_coachingDynamicCallLists]
FOREIGN KEY([sessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingDynamicCallLists] CHECK CONSTRAINT
[FK_coachingDynamicCallLists_coachingDynamicCallLists]
GO

ALTER TABLE [dbo].[coachingSessionActionItems]  WITH CHECK ADD
 CONSTRAINT [FK_coachingSessionActionItems_coachingSessions]
FOREIGN KEY([sessionID])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingSessionActionItems] CHECK CONSTRAINT
[FK_coachingSessionActionItems_coachingSessions]
GO
ALTER TABLE [dbo].[coachingSessionNotes]  WITH CHECK ADD
 CONSTRAINT [FK_coachingSessionsTbl_coachingSessionNotesTbl]
FOREIGN KEY([sessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingSessionNotes] CHECK CONSTRAINT
[FK_coachingSessionsTbl_coachingSessionNotesTbl]
GO
ALTER TABLE [dbo].[coachingSessionResources]  WITH CHECK ADD
 CONSTRAINT [FK_coachingSessionResources_coachingSessions] FOREIGN
KEY([sessionID])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingSessionResources] CHECK CONSTRAINT
[FK_coachingSessionResources_coachingSessions]
GO
```

```
ALTER TABLE [dbo].[coachingStaticCallLists]  WITH CHECK ADD
 CONSTRAINT [FK_coachingStaticCallLists_coachingSessions] FOREIGN
KEY([sessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingStaticCallLists] CHECK CONSTRAINT
[FK_coachingStaticCallLists_coachingSessions]
GO
GO

ALTER TABLE [dbo].[FormsAnsweredTbl]  WITH CHECK ADD  CONSTRAINT
[FK_FormsAnsweredTbl_coachingSessions] FOREIGN
KEY([CoachingSessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[FormsAnsweredTbl] CHECK CONSTRAINT
[FK_FormsAnsweredTbl_coachingSessions]
GO
SET ANSI_NULLS OFF
GO

SET QUOTED_IDENTIFIER OFF
GO

create VIEW [dbo].[userMessagesView]



AS



SELECT     cft.callForwardTaskID AS taskId,
dbo.utc2localTime(dbo.tod2time(cft.creationDate)) AS date,
cft.creator AS 'from', cft.subject, 2 AS type,



                      1 AS subType, '' as permaLink, 0 AS action,
cfts.recipient AS userLogin, cft.callid as callid, cft.externalId
```

```
as externalId

FROM         dbo.callForwardTasksTbl AS cft INNER JOIN

                      dbo.callForwardTasksStatusTbl AS cfts ON
cfts.callForwardTaskID = cft.callForwardTaskID

WHERE     (cfts.notificationMessageDeleted = 0)




UNION



SELECT     sessionId AS taskId, dbo.utc2localTime(timeCreated) AS
date, creator AS 'from', name AS subject, 1 AS type, 0 AS subType,
'' AS permalink, 0 AS action,

                      trainee AS userLogin, -1 as callid, '' as
externalId

FROM         dbo.coachingSessions AS cs
```

```
WHERE     (accepted = 0)




UNION




SELECT     noteId AS taskId, dbo.utc2localTime(creationDate) AS
date, creator AS 'from', subject, 3 AS type, 0 AS subType, '' AS
permalink,

                      0 AS action, recipient AS userLogin, -1 as
callid, '' as externalId

FROM         dbo.notesTbl AS nt

WHERE     (messageNotificationDeleted = 0)


GO

          GO
```

```
IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingResourceViews_coachingSessionResources]')
AND parent_object_id = OBJECT_ID(N'[dbo].[coachingResourceViews]'))
ALTER TABLE [dbo].[coachingResourceViews] DROP CONSTRAINT
[FK_coachingResourceViews_coachingSessionResources]
GO

CREATE TABLE [dbo].[TMP_coachingResourceViews](
        [id] [int] IDENTITY(1,1) NOT NULL,
        [resourceID] [int] NOT NULL,
        [resourceViewer] [Nvarchar](256) NOT NULL,
        [timestamp] [datetime] NOT NULL) ON [PRIMARY]

GO


ALTER TABLE dbo.[TMP_coachingResourceViews] SET (LOCK_ESCALATION =
TABLE)
GO
GO
GRANT SELECT ON dbo.[TMP_coachingResourceViews] TO reports,utopy
 AS dbo
GO
GRANT INSERT ON dbo.[TMP_coachingResourceViews] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[TMP_coachingResourceViews] TO utopy AS dbo
GO
SET IDENTITY_INSERT dbo.[TMP_coachingResourceViews] ON
GO

IF EXISTS(SELECT * FROM dbo.[coachingResourceViews])
         EXEC('INSERT INTO dbo.[TMP_coachingResourceViews]
(Id,resourceID,resourceViewer,timestamp)
                SELECT Id,resourceID, CONVERT(nvarchar(50),
resourceViewer), timestamp FROM dbo.[coachingResourceViews] WITH
(HOLDLOCK TABLOCKX)')
GO

SET IDENTITY_INSERT dbo.[TMP_coachingResourceViews] OFF
```

```
GO

/****** Object:  Table [dbo].[coachingResourceViews]    Script
Date: 09/18/2014 10:54:34 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[coachingResourceViews]') AND type in (N'U'))
DROP TABLE [dbo].[coachingResourceViews]
GO

GO
EXECUTE sp_rename N'dbo.TMP_coachingResourceViews',
N'coachingResourceViews', 'OBJECT'

 ALTER TABLE coachingResourceViews ADD CONSTRAINT
[PK_coachingResourceViews] PRIMARY KEY CLUSTERED
(
        [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
SET ANSI_PADDING ON
GO



IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingSessionResources_coachingSessions]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingSessionResources]'))
ALTER TABLE [dbo].[coachingSessionResources] DROP CONSTRAINT
[FK_coachingSessionResources_coachingSessions]
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO
```

```
CREATE TABLE [dbo].[tmp_coachingSessionResources](
        [resourceID] [int] IDENTITY(1,1) NOT NULL,
        [sessionID] [int] NOT NULL,
        [description] [varchar](256) NOT NULL,
        [resourceType] [int] NOT NULL,
        [linkPath] [varchar](2000) NOT NULL,
        [IsActive] [bit] NOT NULL,
        [creator] [varchar](256) NOT NULL,
        [timeCreated] [datetime] NOT NULL) ON [PRIMARY]

GO
ALTER TABLE dbo.[tmp_coachingSessionResources] SET (LOCK_ESCALATION
= TABLE)
GO
GO
GRANT SELECT ON dbo.[tmp_coachingSessionResources] TO utopy  AS dbo
GO
GRANT INSERT ON dbo.[tmp_coachingSessionResources] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_coachingSessionResources] TO utopy,web AS
dbo
GO
GRANT UPDATE ON dbo.[tmp_coachingSessionResources] TO utopy  AS dbo
GO
SET IDENTITY_INSERT dbo.[tmp_coachingSessionResources] ON
GO

IF EXISTS(SELECT * FROM dbo.[coachingSessionResources])
         EXEC('INSERT INTO dbo.[tmp_coachingSessionResources]
(resourceID,sessionID,description,
resourceType,linkPath,IsActive,creator,timeCreated)
                SELECT resourceID,sessionID, CONVERT(nvarchar(256),
description), resourceType,linkPath,IsActive,
CONVERT(nvarchar(256), creator),timeCreated FROM
dbo.[http://docs.genesys.com/Documentation/SPMI/draft/Upgrade/
tmp_coachingSessionResources tmp_coachingSessionResources] WITH
(HOLDLOCK TABLOCKX)')
GO

SET IDENTITY_INSERT dbo.[tmp_coachingSessionResources] OFF
GO
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
```

```
OBJECT_ID(N'[dbo].[coachingSessionResources]') AND type in (N'U'))
DROP TABLE [dbo].[coachingSessionResources]
GO
EXECUTE sp_rename N'dbo.[tmp_coachingSessionResources]',
N'coachingSessionResources', 'OBJECT'

ALTER TABLE [coachingSessionResources] ADD CONSTRAINT
[PK_coachingSessionResources] PRIMARY KEY CLUSTERED
(
        [resourceID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]

SET ANSI_PADDING ON
GO

ALTER TABLE [dbo].[coachingSessionResources]  WITH CHECK ADD
 CONSTRAINT [FK_coachingSessionResources_coachingSessions] FOREIGN
KEY([sessionID])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO

ALTER TABLE [dbo].[coachingSessionResources] CHECK CONSTRAINT
[FK_coachingSessionResources_coachingSessions]
GO
ALTER TABLE [dbo].[coachingResourceViews]  WITH CHECK ADD
 CONSTRAINT [FK_coachingResourceViews_coachingSessionResources]
FOREIGN KEY([resourceID])
REFERENCES [dbo].[coachingSessionResources] ([resourceID])
GO

ALTER TABLE [dbo].[coachingResourceViews] CHECK CONSTRAINT
[FK_coachingResourceViews_coachingSessionResources]
GO



          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
```

```
OBJECT_ID(N'[dbo].[FK_QMQueuesTbl_searchTbl]') AND parent_object_id
= OBJECT_ID(N'[dbo].[QMQueuesTbl]'))
ALTER TABLE [dbo].[QMQueuesTbl] DROP CONSTRAINT
[FK_QMQueuesTbl_searchTbl]
GO

/****** Object:  Table [dbo].[searchTbl]    Script Date: 09/18/2014
13:16:10 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_searchTbl](
        [searchId] [int] NOT NULL,
        [searchName] [nvarchar](50) NULL,
        [searchFilter] [xml] NOT NULL,
        [stopCondition] [xml] NOT NULL,
        [isActive] [bit] NOT NULL,
        [createdBy] [nvarchar](256) NOT NULL,
        [belongsToGroups] [xml] NOT NULL,
        [sharedWithUsers] [xml] NULL,
        [creationTime] [datetime] NOT NULL,
        [processCalls] [bit] NOT NULL,
        [isDeactivating] [bit] NOT NULL,
        [grammarName] [nvarchar](max) NULL,
        [searchType] [int] NULL) ON [PRIMARY]

GO

SET ANSI_PADDING ON
GO

GO
ALTER TABLE dbo.tmp_searchTbl SET (LOCK_ESCALATION = TABLE)
GO
GO
GRANT SELECT ON dbo.tmp_searchTbl TO utopy,Platform,web  AS dbo
```

```
GO
GRANT INSERT ON dbo.tmp_searchTbl TO Platform,web  AS dbo
GO
GRANT DELETE ON dbo.tmp_searchTbl TO utopy,web,Platform,web  AS dbo
GO
GRANT UPDATE ON dbo.tmp_searchTbl TO Platform ,web AS dbo
go

IF EXISTS(SELECT * FROM dbo.searchTbl)
         EXEC('INSERT INTO dbo.tmp_searchTbl
([searchId],[searchName],[searchFilter],[stopCondition],[isActive],[createdBy
                SELECT [searchId],CONVERT(nvarchar(50),
searchName),[searchFilter],[stopCondition],[isActive],CONVERT(nvarchar(256),
createdBy),[belongsToGroups],[sharedWithUsers],[creationTime],[processCalls],
grammarName),[searchType] FROM dbo.searchTbl WITH (HOLDLOCK
TABLOCKX)')
GO
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[searchTbl]') AND type in (N'U'))
DROP TABLE [dbo].[searchTbl]
GO
EXECUTE sp_rename N'dbo.tmp_searchTbl', N'searchTbl', 'OBJECT'
ALTER TABLE [searchTbl] ADD  CONSTRAINT [PK_searchTbl] PRIMARY KEY
CLUSTERED
(
        [searchId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]

/****** Object:  Table [dbo].[searchTbl]    Script Date: 09/18/2014
13:16:10 ******/

ALTER TABLE [dbo].[QMQueuesTbl]  WITH CHECK ADD  CONSTRAINT
[FK_QMQueuesTbl_searchTbl] FOREIGN KEY([searchID])
REFERENCES [dbo].[searchTbl] ([searchId])
GO

ALTER TABLE [dbo].[QMQueuesTbl] CHECK CONSTRAINT
[FK_QMQueuesTbl_searchTbl]
GO
```

```
          GO

IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_QMquotaParametersTbl_QMParameterTypeTbl]')
AND parent_object_id = OBJECT_ID(N'[dbo].[QMquotaParametersTbl]'))
ALTER TABLE [dbo].[QMquotaParametersTbl] DROP CONSTRAINT
[FK_QMquotaParametersTbl_QMParameterTypeTbl]
GO

/****** Object:  Table [dbo].[QMParameterTypeTbl]    Script Date:
09/18/2014 14:14:11 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_QMParameterTypeTbl](
        [ParameterTypeID] [int] NOT NULL,
        [name] [nvarchar](256) NOT NULL,
        [possibleValues] [xml] NULL,
        [ParameterDisplayType] [int] NOT NULL) ON [PRIMARY]

GO
ALTER TABLE dbo.[tmp_QMParameterTypeTbl] SET (LOCK_ESCALATION =
TABLE)
GO
GRANT SELECT ON dbo.[tmp_QMParameterTypeTbl] TO utopy  AS dbo
GO


IF EXISTS(SELECT * FROM dbo.QMParameterTypeTbl)
         EXEC('INSERT INTO dbo.[tmp_QMParameterTypeTbl]
(ParameterTypeID,[name],[possibleValues],[ParameterDisplayType])
                SELECT ParameterTypeID, CONVERT(nvarchar(256),
name), possibleValues,ParameterDisplayType FROM
dbo.[QMParameterTypeTbl] WITH (HOLDLOCK TABLOCKX)')
GO
```

```
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[QMParameterTypeTbl]') AND type in (N'U'))
DROP TABLE [dbo].[QMParameterTypeTbl]
GO
EXECUTE sp_rename N'dbo.tmp_QMParameterTypeTbl',
N'QMParameterTypeTbl', 'OBJECT'

ALTER TABLE QMParameterTypeTbl ADD CONSTRAINT
[PK_QMQuotaParameterTypeTbl] PRIMARY KEY CLUSTERED
(
        [ParameterTypeID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]

ALTER TABLE [dbo].[QMquotaParametersTbl]  WITH CHECK ADD
 CONSTRAINT [FK_QMquotaParametersTbl_QMParameterTypeTbl] FOREIGN
KEY([parameterType])
REFERENCES [dbo].[QMParameterTypeTbl] ([ParameterTypeID])
GO

ALTER TABLE [dbo].[QMquotaParametersTbl] CHECK CONSTRAINT
[FK_QMquotaParametersTbl_QMParameterTypeTbl]
GO

          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_QMQueuesTbl_searchTbl]') AND parent_object_id
= OBJECT_ID(N'[dbo].[QMQueuesTbl]'))
ALTER TABLE [dbo].[QMQueuesTbl] DROP CONSTRAINT
[FK_QMQueuesTbl_searchTbl]
GO

/****** Object:  Table [dbo].[QMQueuesTbl]    Script Date: 09/22/
2014 09:14:39 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO
```

```
SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_QMQueuesTbl](
        [queueID] [int] IDENTITY(1,1) NOT NULL,
        [name] [nvarchar](200) NOT NULL,
        [description] [nvarchar](1000) NULL,
        [quotaType] [int] NOT NULL,
        [searchID] [int] NOT NULL,
        [isActive] [bit] NOT NULL,
        [creator] [nvarchar](256) NOT NULL,
        [binaryData] [image] NULL

) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
SET IDENTITY_INSERT dbo.[tmp_QMQueuesTbl] ON
GO

ALTER TABLE dbo.[tmp_QMQueuesTbl] SET (LOCK_ESCALATION = TABLE)
GO
GRANT SELECT ON dbo.[tmp_QMQueuesTbl] TO Web,utopy  AS dbo
GO
GRANT INSERT ON dbo.[tmp_QMQueuesTbl] TO utopy  AS dbo
GO
GRANT DELETE ON dbo.[tmp_QMQueuesTbl] TO utopy  AS dbo
GO
GRANT UPDATE ON dbo.[tmp_QMQueuesTbl] TO utopy  AS dbo
GO
GRANT VIEW DEFINITION ON dbo.[tmp_QMQueuesTbl] TO utopy  AS dbo
GO
IF EXISTS(SELECT * FROM [dbo].QMQueuesTbl)
         EXEC('INSERT INTO dbo.[tmp_QMQueuesTbl]
(queueID,[name],[description],[quotaType],[searchID],[isActive],[creator],[bi
                SELECT queueID, CONVERT(nvarchar(200), name),
CONVERT(nvarchar(1000),
description),[quotaType],[searchID],[isActive],CONVERT(nvarchar(1000),
creator),[binaryData] FROM dbo.[QMQueuesTbl] WITH (HOLDLOCK
TABLOCKX)')
GO
GO
SET IDENTITY_INSERT dbo.[tmp_QMQueuesTbl] OFF
GO
```

```
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[QMQueuesTbl]') AND type in (N'U'))
DROP TABLE [dbo].[QMQueuesTbl]
GO
EXECUTE sp_rename N'dbo.tmp_QMQueuesTbl', N'QMQueuesTbl', 'OBJECT'
GO

ALTER TABLE [dbo].[QMQueuesTbl]   ADD CONSTRAINT [PK_QMQueuesTbl]
PRIMARY KEY CLUSTERED
(
        [queueID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
ALTER TABLE [dbo].[QMQueuesTbl]  WITH CHECK ADD  CONSTRAINT
[FK_QMQueuesTbl_searchTbl] FOREIGN KEY([searchID])
REFERENCES [dbo].[searchTbl] ([searchId])
GO

ALTER TABLE [dbo].[QMQueuesTbl] CHECK CONSTRAINT
[FK_QMQueuesTbl_searchTbl]
GO


          GO


IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_QMquotaParametersTbl_QMQuotaTypeTbl]') AND
parent_object_id = OBJECT_ID(N'[dbo].[QMquotaParametersTbl]'))
ALTER TABLE [dbo].[QMquotaParametersTbl] DROP CONSTRAINT
[FK_QMquotaParametersTbl_QMQuotaTypeTbl]
GO
/****** Object:  Table [dbo].[QMQuotaTypeTbl]    Script Date: 09/21/
2014 11:05:41 ******/



/****** Object:  Table [dbo].[QMQuotaTypeTbl]    Script Date: 09/21/
2014 11:05:41 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[tmp_QMQuotaTypeTbl](
        [quotaTypeID] [int] NOT NULL,
        [name] [nvarchar](256) NOT NULL,
        [quotaPhrase] [nvarchar](1000) NOT NULL

) ON [PRIMARY]

GO

SET ANSI_PADDING ON
GO

ALTER TABLE dbo.[tmp_QMQuotaTypeTbl] SET (LOCK_ESCALATION = TABLE)
GO
GRANT SELECT ON dbo.[tmp_QMQuotaTypeTbl] TO utopy  AS dbo
GO


IF EXISTS(SELECT * FROM [dbo].[QMQuotaTypeTbl])
         EXEC('INSERT INTO dbo.[tmp_QMQuotaTypeTbl]
(quotaTypeID,[name],quotaPhrase)
                SELECT quotaTypeID, CONVERT(nvarchar(256), name),
CONVERT(nvarchar(256), quotaPhrase) FROM dbo.[QMQuotaTypeTbl] WITH
(HOLDLOCK TABLOCKX)')
GO
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[QMQuotaTypeTbl]') AND type in (N'U'))
DROP TABLE [dbo].[QMQuotaTypeTbl]
GO
EXECUTE sp_rename N'dbo.tmp_QMQuotaTypeTbl', N'QMQuotaTypeTbl',
'OBJECT'
ALTER TABLE [dbo].QMQuotaTypeTbl add CONSTRAINT [PK_QMQuotaType]
PRIMARY KEY CLUSTERED
(
        [quotaTypeID] ASC
```

```
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]

Go
ALTER TABLE [dbo].[QMquotaParametersTbl]  WITH CHECK ADD
 CONSTRAINT [FK_QMquotaParametersTbl_QMQuotaTypeTbl] FOREIGN
KEY([quotaType])
REFERENCES [dbo].[QMQuotaTypeTbl] ([quotaTypeID])
GO

ALTER TABLE [dbo].[QMquotaParametersTbl] CHECK CONSTRAINT
[FK_QMquotaParametersTbl_QMQuotaTypeTbl]
GO
          GO

update webserviceparams set luceneMaxClauseCount=10000

GO
          GO

ALTER TABLE webServiceParams ADD resetPasswordMailSubject
varchar(max) NOT NULL DEFAULT('Reset SpeechMiner password')
ALTER TABLE webServiceParams ADD resetPasswordMailBody varchar(max)
NOT NULL DEFAULT('Reset your password at <resetLink>')
ALTER TABLE webServiceParams ADD resetPasswordTokenExpirationTime
int NOT NULL DEFAULT(4)
ALTER TABLE webServiceParams ADD PasswordRecovery bit NOT NULL
DEFAULT(0)



SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

CREATE TABLE [dbo].[userResetToken](
```

```
        [UserName] varchar(256) NOT NULL,
        [ResetToken] varchar(256) NOT NULL,
        [expirationTime] int NOT NULL)
GO

SET ANSI_PADDING ON
GO

GRANT UPDATE ON [dbo].[userResetToken] TO Web  AS dbo
GRANT INSERT ON [dbo].[userResetToken] TO Web  AS dbo
GRANT DELETE ON [dbo].[userResetToken] TO Web  AS dbo
GRANT SELECT ON [dbo].[userResetToken] TO Web  AS dbo

          GO

update wildcardGrammars set regularExpression='(?<![\w*])((\d|0\d|
1[0-2]|\*?\*):([0-5]\d|\*\*)(:[0-5]\d|\*\*)?\s?(a|p)m|((\d|0\d|1\d|
2[0-3]|\*?\*):([0-5]\d|\*\*)(:[0-5]\d|\*\*)?(?!(\s?(a|
p)m))))(?![\w*])' where regularExpression is not null and
token='[time]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01])|\*\*)?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(januari|februari|march|april|may|june|juli|august|septemb|
octob|novemb|decemb)\s(of\s)?(19|20|21|\*\*)(\d\d|\*\*))(?![\w*])'
where languageId=0 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillion)?\s(dollar|
cent|buck))(\sand\s(\*|\d){1,2}\scent)?(?![\w*])' where
languageId=0 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((januari|
februari|march|april|may|june|juli|august|septemb|octob|novemb|
decemb)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?|\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|today|tomorrow|yesterday)(?![\w*])'
where languageId=0 and token='[date]'
update wildcardGrammars set regularExpression='\b(si|no|correct)\b'
where languageId=1 and token='[boolean]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
```

```
\*\*)|(ener|febrer|marz|abril|may|juni|juli|agost|septiembr|octubr|
noviembr|diciembr)\s(de\s)?(19|20|21|\*\*)(\d\d|\*\*))(?![\w*])'
where languageId=1 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillon)?\s(dolar|
pes|centav))(\s(y|con)\s(\*|\d){1,2}\scentav)?(?![\w*])' where
languageId=1 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((ener|
febrer|marz|abril|may|juni|juli|agost|septiembr|octubr|noviembr|
diciembr)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|hoy|ma an|ayer)(?![\w*])' where
languageId=1 and token='[date]'
update wildcardGrammars set regularExpression='\b(si|no|correct)\b'
where languageId=2 and token='[boolean]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(ener|febrer|marz|abril|may|juni|juli|agost|septiembr|octubr|
noviembr|diciembr)\s(de\s)?(19|20|21|\*\*)(\d\d|\*\*))(?![\w*])'
where languageId=2 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillon)?\s(eur|
peset|cent))(\s(y|con)\s(\*|\d){1,2}\scent)?(?![\w*])' where
languageId=2 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((ener|
febrer|marz|abril|may|juni|juli|agost|septiembr|octubr|noviembr|
diciembr)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?\*)[-
/.](19|20|21\*\*)?(\d\d|\*\*)|hoy|ma an|ayer)(?![\w*])' where
languageId=2 and token='[date]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(januari|februari|march|april|may|june|juli|august|septemb|
octob|novemb|decemb)\s(of\s)?(19|20|21|\*\*)(\d\d|\*\*))\b' where
languageId=3 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
```

```
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillion)?\s(pound|
penc))(\sand\s(\*|\d){1,2}\spenc)?\b' where languageId=3 and
token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((januari|
februari|march|april|may|june|juli|august|septemb|octob|novemb|
decemb)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|today|tomorrow|yesterday)(?![\w*])'
where languageId=3 and token='[date]'
update wildcardGrammars set regularExpression='\b(oui|non|
correct)\b' where languageId=5 and token='[boolean]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(janvi|f vri|mar|avril|mai|juin|juillet|ao t|septembr|octobr|
novembr|d cembr)\s(de\s)?(19|20|21|\*\*)(\d\d|\*\*))(?![\w*])'
where languageId=5 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|
\d){1,2})?(\smillion)?\s(euro(s?)|dollar|cent))(\set\s(\*|
\d){1,2}\scent)?(?![\w*])' where languageId=5 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((janvi|f
vri|mar|avril|mai|juin|juillet|ao t|septembr|octobr|novembr|d
cembr)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|aujourd''hui|demain|hi)(?![\w*])'
where languageId=5 and token='[date]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|
\d){1,2})?(\smillion)?\s(euro(s?)|dollar|cent))(\sund\s(\*|
\d){1,2}\scent)?(?![\w*])' where languageId=6 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((januar|
februar|marz|april|mai|juno|juni|julei|juli|august|septemb|oktob|
novemb|dezemb)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|
\*\*)(\d\d|\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|
\*?\*)([- /.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|
\*?\*)[- /.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|
```

```
\*\*)?(\d\d|\*\*)|(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|
1[012]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|heut|morg|
gest)(?![\w*])' where languageId=6 and token='[date]'
update wildcardGrammars set regularExpression='\b(sim|n )\b' where
languageId=11 and token='[boolean]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(janeir|fevereir|marc|abril|mai|junh|julh|agost|setembr|
outubr|novembr|dezembr)\s(de\s)?(19|20|21|\*\*)(\d\d|
\*\*))(?![\w*])' where languageId=11 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smilh )?\s(escud|d
l|centav|c nt|c ntim))(\se\s(\*|\d){1,2}\s(centav|c nt|c
ntim))?(?![\w*])' where languageId=11 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((janeir|
fevereir|marc|abril|mai|junh|julh|agost|setembr|outubr|novembr|
dezembr)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|hoj|amanh |ontem)(?![\w*])' where
languageId=11 and token='[date]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smilh )?)(?![\w*])'
where languageId=11 and token='[number]'
update wildcardGrammars set regularExpression='\b(si|no)\b' where
languageId=13 and token='[boolean]'
update wildcardGrammars set regularExpression='(?<![\w*])((0?[1-9]|
1[012]|\*?\*)([- /.](2[89]|3[01]|\*\*))?[- /.](19|20|21|\*\*)?(\d\d|
\*\*)|(januari|februari|march|april|may|june|juli|august|septemb|
octob|novemb|decemb)\s(of\s)?(19|20|21|\*\*)(\d\d|\*\*))(?![\w*])'
where languageId=14 and token='[ccexpdate]'
update wildcardGrammars set regularExpression='(?<![\w*])(\*|
\d){1,9}(\.(\*|\d){1,2})?(\smillion)?\s(dollar|cent|buck)(\sand\
s((\*|\d){1,2})\scent)?(?![\w*])' where languageId=14 and
token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((januari|
februari|march|april|may|june|juli|august|septemb|octob|novemb|
decemb)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
```

```
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?)[- /.](19|
20|21|\*\*)?(\d\d|\*\*)|today|tomorrow|yesterday)(?![\w*])' where
languageId=14 and token='[date]'
update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillion)?\s(dollar|
cent|buck))(\sand\s(\*|\d){1,2}\scent)?)(?![\w*])' where
languageId=16 and token='[currency]'
update wildcardGrammars set regularExpression='(?<![\w*])((januari|
februari|march|april|may|june|juli|august|septemb|octob|novemb|
decemb)\s(0?[1-9]|[12][0-9]|3[01]|\*?\*),?\s(19|20|21|\*\*)(\d\d|
\*\*)|(19|20|21|\*\*)?(\d\d|\*\*)([- /.])(0?[1-9]|1[012]|\*?\*)([-
/.])(0?[1-9]|[12][0-9]|3[01]|\*?\*)|(0?[1-9]|1[012]|\*?\*)[-
/.](0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](19|20|21|\*\*)?(\d\d|\*\*)|
(0?[1-9]|[12][0-9]|3[01]|\*?\*)[- /.](0?[1-9]|1[012]|\*?|\*)[-
/.](19|20|21|\*\*)?(\d\d|\*\*)|today|tomorrow|yesterday)(?![\w*])'
where languageId=16 and token='[date]'

          GO

update wildcardGrammars set regularExpression='(?<![\w*])(-?(\*|
\d){1,3}(,?((\*|\d){3})){0,2}(\.(\*|\d){1,2})?(\smillion)?\s(dollar|
cent|buck))(\sand\s(\*|\d){1,2}\scent)?(?![\w*])' where
languageId=16 and token='[currency]'
          GO

IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingStaticCallLists_coachingSessions]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingStaticCallLists]'))
ALTER TABLE [dbo].[coachingStaticCallLists] DROP CONSTRAINT
[FK_coachingStaticCallLists_coachingSessions]
GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingStaticCallLists_timeCreated]') AND type =
'D')
BEGIN
ALTER TABLE [dbo].[coachingStaticCallLists] DROP CONSTRAINT
[DF_coachingStaticCallLists_timeCreated]
END
```

```
GO

IF  EXISTS (SELECT * FROM dbo.sysobjects WHERE id =
OBJECT_ID(N'[DF_coachingStaticCallLists_timeUpdated]') AND type =
'D')
BEGIN
ALTER TABLE [dbo].[coachingStaticCallLists] DROP CONSTRAINT
[DF_coachingStaticCallLists_timeUpdated]
END

GO

IF  EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id =
OBJECT_ID(N'[dbo].[FK_coachingStaticCallLists_coachingStaticCallListCalls]')
AND parent_object_id =
OBJECT_ID(N'[dbo].[coachingStaticCallListCalls]'))
ALTER TABLE [dbo].[coachingStaticCallListCalls] DROP CONSTRAINT
[FK_coachingStaticCallLists_coachingStaticCallListCalls]
GO

CREATE TABLE [dbo].[tmp_coachingStaticCallLists](
        [listId] [int] IDENTITY(1000,1) NOT NULL,
        [name] [nvarchar](50) NOT NULL,
        [creator] [nvarchar](256) NOT NULL,
        [timeCreated] [datetime] NOT NULL,
        [timeUpdated] [datetime] NOT NULL,
        [sessionId] [int] NULL,
        [BinaryData] [image] NULL,

) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
ALTER TABLE dbo.tmp_coachingStaticCallLists SET (LOCK_ESCALATION =
TABLE)
GO
GO
GRANT SELECT ON dbo.tmp_coachingStaticCallLists TO
Web,reports,utopy  AS dbo
GO
GRANT INSERT ON dbo.tmp_coachingStaticCallLists TO utopy  AS dbo
GO
GRANT DELETE ON dbo.tmp_coachingStaticCallLists TO utopy,web  AS dbo
GO
```

```
GRANT UPDATE ON dbo.tmp_coachingStaticCallLists TO utopy  AS dbo
go
GRANT UPDATE ON dbo.tmp_coachingStaticCallLists(name)  TO web  AS
dbo
GO
SET IDENTITY_INSERT dbo.tmp_coachingStaticCallLists ON
GO

IF EXISTS(SELECT * FROM dbo.coachingStaticCallLists)
         EXEC('INSERT INTO dbo.tmp_coachingStaticCallLists
(listId,name,creator,timeCreated,timeUpdated,sessionId,BinaryData)
               SELECT listId, CONVERT(nvarchar(50), name),
CONVERT(nvarchar(50),
creator),timeCreated,timeUpdated,sessionId,BinaryData FROM
dbo.coachingStaticCallLists WITH (HOLDLOCK TABLOCKX)')
GO

SET IDENTITY_INSERT dbo.tmp_coachingStaticCallLists OFF
GO

/****** Object:  Table [dbo].[coachingStaticCallLists]    Script
Date: 09/18/2014 10:13:42 ******/
IF  EXISTS (SELECT * FROM sys.objects WHERE object_id =
OBJECT_ID(N'[dbo].[coachingStaticCallLists]') AND type in (N'U'))
DROP TABLE [dbo].[coachingStaticCallLists]
GO
EXECUTE sp_rename N'dbo.tmp_coachingStaticCallLists',
N'coachingStaticCallLists', 'OBJECT'

ALTER TABLE coachingStaticCallLists ADD CONSTRAINT
[PK_coachingStaticCallLists] PRIMARY KEY CLUSTERED
(
        [listId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  =
ON) ON [PRIMARY]
ALTER TABLE [dbo].[coachingStaticCallLists]  WITH CHECK ADD
 CONSTRAINT [FK_coachingStaticCallLists_coachingSessions] FOREIGN
KEY([sessionId])
REFERENCES [dbo].[coachingSessions] ([sessionId])
GO
```

```
ALTER TABLE [dbo].[coachingStaticCallLists] CHECK CONSTRAINT
[FK_coachingStaticCallLists_coachingSessions]
GO

ALTER TABLE [dbo].[coachingStaticCallLists] ADD  CONSTRAINT
[DF_coachingStaticCallLists_timeCreated]  DEFAULT (getutcdate())
FOR [timeCreated]
GO

ALTER TABLE [dbo].[coachingStaticCallLists] ADD  CONSTRAINT
[DF_coachingStaticCallLists_timeUpdated]  DEFAULT (getutcdate())
FOR [timeUpdated]
GO


ALTER TABLE [dbo].[coachingStaticCallListCalls]  WITH CHECK ADD
 CONSTRAINT
[FK_coachingStaticCallLists_coachingStaticCallListCalls] FOREIGN
KEY([listId])
REFERENCES [dbo].[coachingStaticCallLists] ([listId])
GO

ALTER TABLE [dbo].[coachingStaticCallListCalls] CHECK CONSTRAINT
[FK_coachingStaticCallLists_coachingStaticCallListCalls]
          GO


ALTER TABLE dbo.callMetaExTypes ADD displayName varchar(256) NULL
GO

-- add permission for the page
INSERT INTO [dbo].[objectPermissionsTbl]
           ([objectId]
           ,[groupId]
           ,[description]
           ,[values]
           ,[configurable]
           ,[explanation])
     VALUES
           (72
           ,13
           ,'Meta Data Manager'
```

```
           ,'<Values xmlns:xsd="http://www.w3.org/2001/
XMLSchema"><Value><display>Show
page</display><value>2</value></Value><Value><display>Hide
page</display><value>0</value></Value></Values>'
           ,1
           ,'')
GO

-- add the permission by default to administrator role
insert into dbo.rolePermissionsTbl values(72,5,2)
GO

GRANT select, insert, update on callMetaExTypes to web
GO
          GO

Update RecognitionLanguages
set MinIndexConfidence=40 where index1=17
GO
          GO

CREATE QUEUE PurgeTaskQueue
    WITH STATUS=ON,
    ACTIVATION (
        PROCEDURE_NAME = sp_runPurgeTaskService,
        MAX_QUEUE_READERS = 1,
        EXECUTE AS OWNER ) ;
GO

CREATE SERVICE PurgeTaskService ON QUEUE PurgeTaskQueue ([DEFAULT]);
GO
          GO

update dbo.versionTbl set version= '8.5.201.252' where resource in
('SM', 'SMART')
go
```

SpeechMiner 8.5.2 to 8.5.3 Upgrade Procedure

This document explains how to upgrade SpeechMiner from version 8.5.2 to version 8.5.3

Pre-upgrade Requirements

- Request the most recent release of the 8.5.3 software from your Genesys representative.
- Request the SpeechMiner license from Genesys Licensing.
- Verify that the following are installed:
  - Microsoft .NET Framework 4.5 SP1 (4.5.1) must be installed on all machines that will run SpeechMiner components or interact with SpeechMiner.
    You can download the installation package at:
    http://www.microsoft.com/en-us/download/details.aspx?id=40773.
  - Microsoft Visual C++ 2013 Redistributable must be installed on all machines that will run SpeechMiner components or interact with SpeechMiner.
    You can download the installation package at:
    http://www.microsoft.com/en-us/download/details.aspx?id=40784.

Upgrade Checklist

The following checklist summarizes all the procedures required for upgrading SpeechMiner. Make sure to complete all of the required procedures.

Item to Check	Details
Storage Requirements	To successfully complete the upgrade process, the data partition in the SQL server must have available disk space. The minumum required storage for the upgrade should be twice the size of the production database .mdf file.
Perform Pre-upgrade Tests	Since upgrading SpeechMiner can take several hours it is important to avoid delays. For this reason, it is recommended to test SMUpgrade before you begin the upgrade procedure. To test SMUpgrade, perform one of the following: • Create a copy of the database and send it to your Genesys counterpart. Genesys will test the SMUpgrade process in a lab environment.

	• Create a copy of the database on a separate SQL server. Provide Genesys with the details of the SQL server and Genesys will test the SMUpgarde process on this temporary server.
Check for Customization	If any customizations were implemented on your DB, make sure that they are part of the new version, or that they can be used in the new version without changes. Contact Genesys Customer Care for assistance.
Purging Old Data	Most systems have a data retention policy in place. Data (for example, audio,exploration data, etc.) that is older than the specified period of time is automatically deleted. If you do not have a data retention policy in place, it is recommended to determine what data should be saved and what can be discarded, since deleting the data before the upgrade will reduce both the time it takes to run the upgrade process and the storage-space requirements.

Rollback Plan

To ensure that you can revert back to SpeechMiner 8.5.2, keep the 8.5.2 DB active on the server, rather than just keeping a backup file.

Do not delete the 8.5.2 data folders (`index`, `grammars`, etc.). Instead, configure the 8.5.2 system with new data folders. Create the following new folders to ensure that you will not lose 8.5.2 data:

- Create the following empty folders:
  - Input
  - Interaction Receiver Input
  - Filtered
  - Index (this folder will be populated during the upgrade procedure).
- Copy the content of the following existing folders to new folders with the same name:
  - Store
  - Grammar
  - Backup

For detailed information about the folders you should create, refer to Required Folders

Since the 8.5.2 DB and data folders are saved and available, back-out steps are not required if problems arise with the upgrade process before you uninstall 8.5.2. The 8.5.2 system should still be configured and functional.

After you uninstall SpeechMiner 8.5.2 and install SpeechMiner 8.5.3, the only way to revert back to 8.5.2 is to install 8.5.2 again and update the config files using SMConfig. However, since the DB and data folders would not have been deleted, they should be available and ready to use without changing the system configuration.

Upgrade Procedure

The following table lists the approximate times required to complete the upgrade steps:

Step	Time
Stop the system (step 1)	15 minutes
Backup the database (step 2)	120 minutes
Create target database (step 3)	30 minutes
Run SMUpgrade (steps 6 to 13)	10 to 20 hours
Configuring and starting the system (steps 14 to end)	60 minutes

1. Using `SMConfig->Services->Stop Services`, stop the 8.5.2 system.
2. Create a copy of the source DB and upgrade it to the latest build:
   The source DB must be in build 8.5.201. Refer to the `versionTbl` table to determine the correct version.
   If you have a build that is later than 8.5.201.257, contact Genesys Customer Care.
   If the source DB is not the latest build and you do not want to update it to the latest build, create a copy of the source DB and update the copy to the latest build.

   > **Important**
   >
   > These steps are necessary because the 8.5.2 DB schema needs to be updated to the latest schema in order for the rest of the upgrade process to succeed.

   Use the copy of the source DB as the baseline for the 8.5.3 upgrade.

   a. Back up the 8.5.2 index folder to a backup folder (see Configuring SpeechMiner—Index).
   b. Create a copy of the source DB (back up the DB and then restore it in another location).

c. Configure the copy of the DB with the Index backup. This will be the baseline for the upgrade.
d. Implement the schema changes on the baseline DB to bring the schema into line with the latest 8.5.2 build version. For this step you will need assistance from Customer Care.

3. Create the 8.5.3 target DB as follows:
   - Manually—Refer to **Installing the SpeechMiner Database > Manual** tab.

   Or
   - Setup Wizard— Refer to **Installing the SpeechMiner Database > Setup Wizard** tab.
4. If the MS-SQL server is an Enterprise Edition, run `EXEC sp_create_DB_storage_partitions` on the target database.
5. If your source and target databases are on different servers, make sure the servers are linked in both directions, using the stored procedures `sp_addlinkedserver` and `sp_addlinkedsrvlogin`, as needed.
6. Install and run SMUpgrade (to migrate the data from the 8.5.2 DB to the 8.5.3 DB), as follows:

Prerequisites:

- When migrating a large database, make sure that the hard drive that hosts the target database has enough storage space.

Usage

a. Query the `versionTbl` table to ensure that your 8.5.2 source database is updated to the latest 8.5.2 schema.
b. Verify that your recovery model is either `Simple` or `Bulk-logged`. To determine which recovery model you have, right click `db > properties > options > recovery model`.
c. Use the SpeechMiner Installer to install the `SMUpgrade` component. It is recommended to install and run the SMUpgrade component on the SQL server.
d. Configure the following in the `\utopy\tools\bin\release\SMUpgrade.exe.config` file:
   - file locations
   - tables to skip (comma separated list)

- number of threads running concurrently on a large table
- bulk copy usage

It is recommended that the `bulk-load` folder be located on the SQL server. The `bulk-load` folder must be shared and the user running the SQL server service must have full control over it.
If you do not set the bulk copy usage, the `callAudioTbl` upgrade will take up to two times longer on a large DB.
Only use the `skip-tables` configuration if specifically requested by Genesys Customer Care.

```
 <appSettings>
<add key="ErrorLogFile"
value=".\SMUpgradeLog.txt" />
<add key="LogFile"
value=".\tableLog.txt" >
<add key="TimingsFile"
value=".\tableTimings.txt" />
<add key="SkipTables"
value="callTasksTbl" />
<add key="NumThreads" value="10" />

</appSettings>
```

e. Run `SMUpgrade.exe`.
Log in and select the appropriate 8.5.2 source and 8.5.3 destination databases.
The databases that appear in the old databases drop down list include ver8_5_2 in their file name. The databases in the new databases drop down list, include ver8_5_3 in their file name. You can also type relevant databases that are named differently.

Important

It is highly recommended to use the `sa` credentials or a user account with bulk insert permissions.

The user account must belong to the `db_owner` role in the target database. By default, the `DBUser` does not include the `db_owner` role.

f. The GUI shades the tables as follows:
   - Green—The table is finished. Both the source and the target DBs contain the same number of records. This conclusively shows that either the data has been copied or there is no data to copy.
   - Yellow—The number of records in the source and target DBs is not indicative of whether the data in both DBs is identical or not. It is therefore not known whether the table is finished or not.
   - Red—The table is not finished. The number of records in the source and target databases are not the same, and indicating that the data has not been copied in full.

g. Click `Full Upgrade` to run the upgrade, or `Resume Last` if your previous upgrade was interrupted.

   Resuming the last upgrade will shorten the time to run, but might cause problems. To avoid such problems, restore the database again and run the full upgrade. You can also define tables to skip in the configuration file. Every step in the upgrade process is shown in the

GUI.
You can stop the upgrade by clicking the `Close` button. You will be prompted to confirm your action. Note that the window closes immediately, but the process still runs for a while, as it needs to re-enable the indexes it disables when it starts running.
The time each step took is written to the `TimingsFile`. The location of this file is defined in the configuration file.
IN CASE OF FAILURE: Review all status, error and exception notifications in the `ErrorLogFile`.

h. Continue with the upgrade instructions below.

7. If the SpeechMiner Maintenance Job exists, and the `Update time table` step is included, delete the `Update time table` step. Make sure the last step in the job is set to quit the job upon both success and failure.
8. Optional: Uninstall 8.5.2 from all servers. The two versions (8.5.2 and 8.5.3) cannot be running side by side at the same time. Only one version can be registered as the active SpeechMiner service on each server. The installation binaries can be left on the server.
9. Update Microsoft .NET Framework.
10. Update Microsoft C++ Redistributable.
11. Install the 8.5.3 platform on all servers.
12. Install 8.5.3 Web on the Web server.
13. Install 8.5.3 SMART on users' desktops, as required.
14. Deploy SQLCLR on the DB server. Via SQL management run the commands that are in C:\Program Files (x86)\Genesys\Software\Support\sqlclr.sql in the SpeechMiner 8.5.3 database.
15. Update the package folders with the 8.5.3 `.gram` files. The `.gram` files are located in the `<Installation Folder>/Support/Grammars`.
    Alternatively, if you have not made changes to any file in these folders, you can delete their content completely. SMConfig will copy the grammar files to `<Installation folder>/Support/Grammars`.
16. Manually copy the files in `<Installation Folder>\Support\Grammars\Confidence` to the `Global Packages` folder.
17. Run `SMConfig`.

Important

If your target database was restored from a backup file, you may need to "fix" an orphan dbuser. To do this, simply run EXEC sp_change_users_login 'Auto_Fix', 'dbuser'.

a. Configure the Sites & Machines, panel as necessary, and save the changes. Make sure you save this panel even if you have not made any changes.
b. Configure the Services panel and save the changes. Do not start any of the services.
c. Configure the Index panel and save the changes.
d. Update the SpeechMiner License with the new 8.5.2 licenses provided by Genesys Licensing.
e. In the Reports panel, update the `MRSLibrary.dll` on the report server.
f. Deploy the reports to the report server.
g. Start SMART and perform the following:
    a. Right-click on each active Program icon and choose `Activate program`. This will mark all the Programs, Topics, and Categories as changed.
    b. Click the `Apply` button.
    c. In the new `Apply` popup window, choose `Apply all`.
    d. Click the `Apply` button.
h. Using `SMConfig`, start the `UPlatform` services on all the servers.
i. Update the Stored Procedures by copying any existing custom Stored Procedures from the 8.5.2 DB to the 8.5.3 DB.
It is not necessary to copy Stored Procedures that are used by gauges, and are in the `GaugeWidgetProcedures` table, because they are copied automatically.
j. Open the SpeechMiner web-based interface and test the functionality.
k. Update the Database Jobs:
    - All database jobs that point to the 8.5.2 DB should be changed to point to the new 8.5.3 DB. Examples of DB jobs that might need to be changed:
        - DB maintenance job
        - `sp_agentFilterCleanByDays`
        - `sp_updateUntilYesterdayMaxChannels`

    To change a DB job, we recommend that you edit the Job Step property using the SQL Management studio.

l. In the SpeechMiner web-based interface, manually reschedule 8.5.2 reports that should continue to run on a scheduled basis.