

Perceptive Content Database Upgrade Package for SQL Server

Reference Guide

Version: Foundation EP4

Written by: Product Knowledge, R&D
Date: June 2021

Copyright

Information in this document is subject to change without notice. The software described in this document is furnished only under a separate license agreement and may be used or copied only according to the terms of such agreement. It is against the law to copy the software except as specifically allowed in the license agreement. This document or accompanying materials contains certain information which is confidential information of Hyland Software, Inc. and its affiliates, and which is subject to the confidentiality provisions agreed to by you.

All data, names, and formats used in this document's examples are fictitious unless noted otherwise. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright law, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Hyland Software, Inc. or one of its affiliates.

Hyland® and Hyland Software®, as well as Hyland product names, are registered and/or unregistered trademarks of Hyland Software, Inc. and its affiliates in the United States and other countries. All other trademarks, service marks, trade names and products of other companies are the property of their respective owners.

© 2021 Hyland Software, Inc. and its affiliates. All rights reserved.

Table of Contents

Copyright	2
Overview	5
Benefits	5
Summary of steps	6
With uptime.....	6
<i>Setup steps</i>	6
<i>Uptime steps</i>	6
<i>Synchronization</i>	6
<i>Downtime steps</i>	6
Without uptime.....	7
<i>Setup steps</i>	7
<i>Downtime steps</i>	7
Status commands	7
Prerequisites	7
Database Settings for Filtered Indexes	7
<i>Database Settings</i>	7
<i>ODBC Driver (DSN) Settings</i>	7
Database Storage for the IN_INSTANCE_PROP indexes.....	9
Database Storage for the IN_DOC table and indexes	9
SQLCMD Mode.....	10
Create the packages	10
IN_DB_UTIL package	11
IN_DB_UPGRADE package.....	11
Setup steps	12
IN_DB_UPGRADE_SP_UPGRADE_SETUP.....	12
Uptime steps	12
IN_DB_UPGRADE_SP_UPTIME_STEPS.....	13
IN_DB_UPGRADE_SP_UPTIME_GET_STATUS	13
IN_DB_UTIL_SP_SESSIONS_DISPLAY.....	13
Synchronization steps	14
IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC	14
IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC.....	14
IN_DB_UPGRADE_SP_SYNC_GET_STATUS.....	14

Downtime steps	15
Pre-downtime steps	15
Downtime steps	15
<i>IN_DB_UPGRADE_SP_DOWNTIME_STEPS</i>	15
Post downtime steps	15
Additional synchronization commands	15
IN_DB_UPGRADE_SP_SYNC_RESET	16
IN_DB_UPGRADE_SP_SYNC_REMOVE.....	16
Available parameters	16
Parameters	16
Examples.....	18
View and update Upgrade parameter default values	19
View and update Synchronization parameter default values	19
Troubleshooting commands	20
Cleaning up after an early exit	20
Debugging	21
Removing the Upgrade Package	21

Overview

The following document guides you through the database upgrade process using a collection of procedures and functions collectively referred to as a package. There are two packages, IN_DB_UTIL and IN_DB_UPGRADE.

You can use the IN_DB_UPGRADE package as an alternative to the traditional database incremental scripts.

The IN_DB_UPGRADE package provides a means for you to upgrade the Perceptive Content database schema in a manner that facilitates the execution of the most time-consuming schema changes in uptime to reduce the duration of downtime events during database upgrades.

This package supports upgrading the inuser database schema to version 7.7.0.0 (EP3) or 7.5.0.0 (EP2) from 7.5.0.0 or 7.4.0.3 or 7.2.3.0 or 7.2.0.0 or 7.1.5.2 or 7.1.4.0 or 7.1.3.3. It supports both the Unicode and ANSI versions of the schema.

This upgrade package is ideal for customers with very large databases who might benefit from executing the longest running operations in uptime, resulting in a much shorter downtime event during the upgrade.

This upgrade is also ideal for customers that are many versions behind as it simplifies the upgrade process by facilitating the upgrade with just a few commands instead of manually executing multiple incremental DDL scripts to upgrade the database schema to the latest version.

Using this package to upgrade the database schema involves the duplication of the IN_DOC table as well as several large indexes. The size of the indexes depends on your schema version. This requires the allocation of additional storage for the duration of the upgrade, which can be reclaimed afterwards if necessary.

Benefits

- This upgrade package provides for additional flexibility by facilitating an upgrade to 7.7.0.0 or 7.5.0.0 from any previous version between 7.1.3.3 – 7.5.0.0.
- This upgrade package simplifies the upgrade process by facilitating the upgrade with just a few commands instead of manually executing multiple incremental DDL scripts, in the proper order, to upgrade the database schema to the latest version.
- The 7.2.2.0 and 7.5.0.0 incremental database scripts include operations that rebuild the IN_DOC table and its indexes and constraints. These operations are time consuming and result in an extended downtime for larger databases. This upgrade package provides a means for accomplishing these lengthy operations during uptime in the days or hours leading up to the downtime and results in a much shorter downtime event.
- The 7.4.0.3 incremental database script includes the rebuild of the indexes on the IN_INSTANCE_PROP table to create them as filtered indexes to improve performance and decrease the size of those indexes. This upgrade package provides a means for accomplishing these index rebuilds in uptime in the days or hours leading up to the downtime event and results in a much shorter downtime event.
- This upgrade package provides you with more control with regard to incrementally completing the qualifying uptime steps over shorter increments of time using the MAX_MINUTES parameter to help limit the number of uptime steps completed whenever executing the IN_DB_UPGRADE_SP_UPTIME_STEPS procedure.

- This upgrade package provides you with more control over database upgrade operations by surfacing parameters such as filegroup names, max degree of parallelism (MAXDOP) and fillfactor for uptime and downtime operations.

Summary of steps

The following section outlines the summary of steps described in this document.

Note Please review the sections for each command, later in this document, for additional details on available parameters for each procedure which can be used for greater control over the upgrade.

With uptime

Setup steps

Use the following command to create and populate the IN_UPGRADE_CONTROL table. For upgrades with a schema version less than 7.5.0.0 this will also create and populate the IN_SYNC_CONTROL table.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPGRADE_SETUP;
```

Uptime steps

Use the following command to execute qualifying uptime schema changes until all steps have completed.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPTIME_STEPS;
```

Use the following command to view the progress of qualifying uptime schema changes.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPTIME_GET_STATUS;
```

Synchronization

For upgrades with a schema version less than 7.5.0.0, use the following commands to setup the synchronization framework. When the uptime steps gets to the 7.4.0.3 to 7.5.0.0 portion of the uptime steps it will automatically run the IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC procedure to setup the synchronization framework for the IN_DOC table. You can, however, manually execute this procedure before then if you want to setup the synchronization ahead of time. Note that it creates overhead as any changes to the IN_DOC table are staged and must be propagated.

```
-- Setup the synchronization framework
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC;

-- Propagate staged data from IN_DOC into NEW_DOC
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC;

-- Synchronization Status Report
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_GET_STATUS 'inuser.IN_DOC';
```

Downtime steps

Use the following command to execute qualifying downtime schema changes.

```
EXEC inuser.IN_DB_UPGRADE_SP_DOWNTIME_STEPS;
```

Without uptime

Setup steps

Use the following command to create the IN_UPGRADE_CONTROL and IN_SYNC_CONTROL tables.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPGRADE_SETUP;
```

Downtime steps

Use the following procedure to execute qualifying uptime steps, synchronization steps, and downtime schema changes.

```
EXEC inuser.IN_DB_UPGRADE_SP_DOWNTIME_STEPS;
```

Status commands

Use the following procedures to check the status of uptime and synchronization steps.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPTIME_GET_STATUS;  
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_GET_STATUS 'inuser.IN_DOC';
```

Use the following procedure to view active session details related to the upgrade or synchronization.

```
EXEC inuser.IN_DB_UTIL_SP_SESSIONS_DISPLAY;
```

Prerequisites

Database Settings for Filtered Indexes

The following applies to upgrades starting on schema versions prior to 7.4.0.3.

Database Settings

The upgrade package will check the following database options and try to change them as follows, if necessary. This is required in order to create filtered indexes.

- SET ANSI_NULLS ON
- SET ANSI_PADDING ON
- SET ANSI_WARNINGS ON
- SET ARITHABORT ON
- SET NUMERIC_ROUNDABORT OFF
- SET QUOTED_IDENTIFIER ON
- SET CONCAT_NULL_YIELDS_NULL ON

ODBC Driver (DSN) Settings

Before executing the IN_DB_UPGRADE_SP_UPTIME_STEPS procedure, you must make the following changes to the Perceptive Content SQL Server ODBC Data Source (DSN) on the Perceptive Content application server(s). Failing to adjust the ODBC options as follows will result in an unplanned downtime.

In order for the Perceptive Content application to be compatible with the new filtered indexes, the “Quoted Identifiers” option must be enabled prior to creating the new filtered indexes. The filtered indexes are created as part of the 7.2.3.0 to 7.4.0.3 uptime schema changes.

After making the changes to the ODBC DSN you will need restart the Perceptive Content services so that all database connections are re-established with these options enabled.

Note Starting with Perceptive Content version 7.3, the native ODBC drivers for SQL Server will replace the DataDirect ODBC drivers. Please reference the Perceptive Content Database Driver Installation and Configuration Guide for version 7.3 for more details.

These manual steps can be disregarded if the following is true:

- If the Perceptive Content Application Server is currently on version 7.3 and you have already switched over to the native ODBC drivers for SQL Server.
- If the Perceptive Content Application Server is currently on version 7.2.3 and you plan to run this script ONLY during downtime (no uptime steps).

These manual steps must be executed if the following is true:

- If the Perceptive Content Application Server is currently on version 7.2.3 and you plan to run this script using the uptime features. If this is the case then please follow the instructions below for the manual steps that must be executed prior to running the uptime steps.

Manual Steps

Execute the following steps within the System DSN tab of the ODBC Data Source Administrator on the Perceptive Content application server(s).

Note If you are running 32-bit and 64-bit agents on the same machine, you must configure the 32-bit and 64-bit ODBC datasources with the same name.

1. Ensure the following options are checked/enabled in the Advanced tab of the Perceptive Content ODBC SQL Server Wire Protocol Driver.
 - "Enable Quoted Identifiers"
 - "AnsiNPW" (This should already be enabled by default)
2. Restart the Perceptive Content services so that all database connections are re-established with these options enabled.

WARNING

Executing the IN_DB_UPGRADE_SP_UPTIME_STEPS procedure to create the new filtered indexes without enabling the Quoted Identifiers option will result in not being able to create new documents.

Note If this happens you will encounter errors in your application server logs similar to the following:

```
error (cat=54HT8B id=54MW2Y loc=38QGJ7 ref=EB6CF6): The database is reporting an error. Contact your ImageNow administrator.
```

```
QUERY: INSERT INTO IN_INSTANCE_PROP(INSTANCE_ID, PROP_ID, IS_NULL, STRING_VAL, NUMBER_VAL, TIME_VAL) VALUES (?, ?, ?, ?, ?, ?)
```

```
STATE:          HY000
NATIVE CODE:    1934
MESSAGE:        [ImageNow][ODBC SQL Server Wire Protocol driver][Microsoft SQL
Server]INSERT failed because the following SET options have incorrect settings:
'QUOTED_IDENTIFIER'.
                Verify that SET options are correct for use with indexed views and/or
indexes on computed columns and/or filtered indexes and/or query notifications and/or
XML data type methods and/or spatial index operations.
```

Database Storage for the IN_INSTANCE_PROP indexes

The following applies to upgrades starting on schema versions prior to 7.4.0.3.

To accommodate the uptime creation of the new filtered indexes on the IN_INSTANCE_PROP table, additional storage will need to be allocated to the respective database filegroup. The new filtered indexes will be created in addition to the existing indexes to ensure proper index coverage. Once all the new indexes have been created, the existing non-filtered indexes will be automatically dropped.

You can use SQL Server Management Studio (SSMS) to get the current size of the IN_INSTANCE_PROP indexes (Index space) by completing the following steps.

- Within SSMS, follow these steps to find the minimum amount of storage required for the upgrade.
 - Within the Object Explorer window, navigate to the IN_INSTANCE_PROP table.
 - Right click on the IN_INSTANCE_PROP table and select Properties.
 - Select the Storage page.
 - Under the General section, you will find the current size in MB.
 - Index space will reflect total size of all the non-clustered indexes on the IN_INSTANCE_PROP table.
- Additional storage considerations
 - The Index space collected above represents the current size of the indexes. Consideration should also be given for storage to accommodate the expected volume of new documents or custom properties created during the period of time when the new indexes are being created and before the existing indexes are dropped.
 - In a default implementation, all the non-clustered indexes are located in the SECONDARY filegroup.

Database Storage for the IN_DOC table and indexes

The following applies to upgrades starting on schema versions prior to 7.5.0.0.

To facilitate the process of upgrading the IN_DOC table in uptime, a copy of the IN_DOC table is created during setup and is synchronized with the original table until the downtime steps have completed. This requires that additional storage be allocated to the database filegroups to accommodate the copy of the IN_DOC table and its indexes.

Before executing the UPGRADE and SYNC setup commands you will want to ensure that you have sufficient free space in the database filegroups where the new version of the IN_DOC table and indexes will be created. You can use SQL Server Management Studio (SSMS) to get the current size of the IN_DOC table (Data space) and its indexes (Index space) by completing the following steps.

- Within SSMS, follow these steps to find the minimum amount of storage required for the upgrade.
 - Within the Object Explorer window, navigate to the IN_DOC table.
 - Right click on the IN_DOC table and select Properties.
 - Select the Storage page.
 - Under the General section, you will find the current size in MB.
 - Data space will reflect the current size of the IN_DOC table (PK_DOC clustered index)
 - Index space will reflect total size of all the non-clustered indexes on the IN_DOC table.
- Additional storage considerations
 - The Data and Index space collected above represents the current size. Consideration should also be given to additional storage to accommodate the expected volume of new documents created during the period of time that the synchronization process is in place.
 - If you are currently using the default FILLFACTOR of 100% (also represented as 0) and planning to create the new indexes using a non-default FILLFACTOR size (<100%) then you will need to add a proportionate amount of storage to account for the extra pages required to hold the same amount of data. For example, if you set fillfactor to 95%, resulting in leaving 5% of each page empty and free to accommodate future data changes, then you will also want to increase the total size allocated by an additional 5% to account for the extra pages that would be required to store the same amount of data.
 - During the downtime steps, the original IN_DOC table and indexes will be replaced by the new versions created during the uptime steps. Upon completion of the upgrade downtime steps and after the original IN_DOC table and its indexes are dropped the storage it consumed will be released and become available within the respective filegroups.
 - In a default implementation, the table (clustered index) will be located in the PRIMARY filegroup and all the non-clustered indexes will be located in the SECONDARY filegroup.
 - In order to reclaim the additional disk space added during the upgrade you may choose to create the new IN_DOC table and indexes in a new, temporary filegroup during the upgrade and move them at a later time, after the upgrade, using the online index rebuild process. Note that online index rebuilds is an Enterprise Edition feature. Once the table (clustered index) and the non-clustered indexes have been moved back to their original filegroups, and the temporary filegroup is empty, you can drop the empty filegroup and its files to reclaim that disk space.

SQLCMD Mode

The following scripts, which create the IN_DB_UTIL and IN_DB_UPGRADE packages, must be executed with SQLCMD Mode enabled. To enable SQLCMD mode for each session, from the main **SSMS** toolbar, click **Query** and then select **SQLCMD Mode**.

Create the packages

Within SQL Server Management Studio (SSMS), execute the following scripts to create the packages.

IN_DB_UTIL package

The IN_DB_UTIL package is a collection of integrated procedures and functions that perform common database tasks that were previously part of the database incremental scripts. The IN_DB_UPGRADE package depends on the IN_DB_UTIL package.

To create the various IN_DB_UTIL database functions and stored procedures used by the upgrade package, complete the following step.

Note Make sure you are connected to the correct database and the v_database_name variable matches the database name you are upgrading.

```
-- Name of Database
:setvar v_database_name 'INOW'
```

- In SSMS, execute the following script.

```
PerceptiveContentDB_IN_DB_UTIL_Package_SQLServer_v4.0.sql
```

IN_DB_UPGRADE package

The IN_DB_UPGRADE package is a collection of integrated procedures and functions that manage the synchronization process and execute database schema changes for both uptime and downtime steps.

Prior to executing this script please review the upgrade and synchronization parameter default values below. These values are saved during execution of the IN_DB_UPGRADE_SP_UPGRADE_SETUP procedure and are used when populating the IN_UPGRADE_CONTROL and IN_SYNC_CONTROL tables. These default values are used by the various procedures unless otherwise specified during execution.

Please reference [the IN_DB_UPGRADE_SP_UPGRADE_SETUP usage notes and parameters](#) section for more details.

Note Make sure you are connected to the correct database and the v_database_name variable matches the database name you are upgrading.

```
-- Name of Database being Upgraded
:setvar v_database_name 'INOW'

-- Target Schema Version
:setvar v_target_schema_version '7.7.0.0'

-- Upgrade Default Values
:setvar v_max_minutes      0
:setvar v_max_dop          0
:setvar v_fillfactor       90
:setvar v_filegroup_name   'PRIMARY'
:setvar v_debug            'NO'

-- Additional Synchronization Default Values (7.5.0.0 and under)
:setvar v_sync_batch_size  50000
:setvar v_sync_defer_fk    'NO'
:setvar v_sync_keep_src    'YES'
```

To create the various database functions and stored procedures used to upgrade the INOW schema to version 7.7.0.0 or 7.5.0.0 from schema versions 7.1.3.3 through 7.5.0.0, complete the following step.

- In SSMS, execute the following script.

```
PerceptiveContentDB_Upgrade_Package_SQLServer_7700.sql
```

Setup steps

To prepare the database for the upgrade, complete the following step.

IN_DB_UPGRADE_SP_UPGRADE_SETUP

The IN_DB_UPGRADE_SP_UPGRADE_SETUP procedure will create the IN_UPGRADE_CONTROL table which holds the default parameter values that are used if not otherwise specified when executing the various upgrade procedures.

Note For instructions on changing the default upgrade parameters after setup, please refer to the [View and update Upgrade parameter default values](#) section.

For upgrades where the schema version is less than 7.5.0.0 this procedure will also create the IN_SYNC_CONTROL table which holds the default parameter values that are used if not otherwise specified when executing for the various synchronization procedures.

Note For instructions on changing the default synchronization parameters after setup, please refer to the [View and update Synchronization parameter default values](#) section.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPGRADE_SETUP;
```

The following parameters are available to use.

- @TARGET_SCHEMA_VERSION
- @FILEGROUP_NAME
- @DEFAULT_MAX_MINUTES
- @DEFAULT_MAX_DOP
- @DEFAULT_FILLFACTOR
- @DEFAULT_BATCH_SIZE
- @SYNC_DEFER_FK_CHECK
- @SYNC_KEEP_SRC_TABLE

Uptime steps

To execute qualifying uptime schema changes, complete the following steps.

IN_DB_UPGRADE_SP_UPTIME_STEPS

The IN_DB_UPGRADE_SP_UPTIME_STEPS procedure facilitates the execution of schema upgrade steps that qualify for execution in uptime. The steps that are executed depends on the starting schema version and the specified target schema version.

You can schedule or manually execute the IN_DB_UPGRADE_SP_UPTIME_STEPS procedure to incrementally execute the uptime steps at any point after upgrade setup and prior to the execution of the downtime steps.

The following schema upgrades currently include qualifying uptime steps:

- IN_DB_UPGRADE_SP_SCHEMA_UPGRADE_7220_7230
 - Creation of the PHSOB_IDX10 index on the IN_PHSOB table
- IN_DB_UPGRADE_SP_SCHEMA_UPGRADE_7230_7403
 - Creation of the filtered indexes on the IN_INSTANCE_PROP table
- IN_DB_UPGRADE_SP_SCHEMA_UPGRADE_7403_7500
 - Rebuild of the IN_DOC table and its indexes

If you choose not to execute any steps in uptime, you can skip this step and go directly to the IN_DB_UPGRADE_SP_DOWNTIME_STEPS procedure. Doing so does not eliminate any steps since the DOWNTIME_STEPS procedure will call the UPTIME_STEPS procedure to ensure all qualifying steps are completed.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPTIME_STEPS;
```

The following parameters are available to use.

- @MAX_MINUTES
- @MAX_DOP
- @FILLFACTOR

IN_DB_UPGRADE_SP_UPTIME_GET_STATUS

The IN_DB_UPGRADE_SP_UPTIME_GET_STATUS procedure will display the overall progress of the qualifying uptime steps.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_UPTIME_GET_STATUS;
```

IN_DB_UTIL_SP_SESSIONS_DISPLAY

The IN_DB_UTIL_SP_SESSIONS_DISPLAY procedure will display the session details associated with any upgrade or synchronization activity that is currently executing.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UTIL_SP_SESSIONS_DISPLAY;
```

Synchronization steps

For upgrades where the starting schema version is less than 7.5.0.0, the synchronization framework is used for propagating data between the IN_DOC and NEW_DOC tables. This framework facilitates the column modifications and population and index creation of the new version of the IN_DOC table in uptime.

When the uptime steps get to the 7.4.0.3 to 7.5.0.0 portion of the upgrade it will automatically run the IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC procedure to setup the synchronization framework for the IN_DOC table.

Note Synchronization creates a small amount of overhead. As data changes in the IN_DOC table, those changes are recorded and staged and must be propagated to the new table. It is advised to delay synchronization until closer to the downtime event.

IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC

The following procedure is executed automatically during the uptime steps but can also be ran manually any time after running upgrade setup to preemptively setup the synchronization framework.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC;
```

The following parameter is available to use.

- @MAX_MINUTES
- @MAX_DOP
- @BATCH_SIZE
- @DEFER_FK_CHECK
- @KEEP_SRC_TABLE
- @FILEGROUP_NAME

IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC

The following procedure synchronizes the new version of the IN_DOC table (NEW_DOC) with the existing version (IN_DOC) based on the rows in the staging table (SYNC_STAGE_DOC).

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC;
```

The following parameters are available to use.

- @MAX_MINUTES
- @BATCH_SIZE

IN_DB_UPGRADE_SP_SYNC_GET_STATUS

The following procedure displays the current status of the synchronization between the IN_DOC and NEW_DOC tables. When executed it displays a status report that includes synchronization metrics, history and details regarding staged data.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_GET_STATUS 'inuser.IN_DOC';
```

The following parameter is required.

- @TABLE_NAME

Downtime steps

Pre-downtime steps

Important Before executing the downtime steps, ensure the following:

- Make sure you have a good full database backup prior to executing the downtime steps.
- Stop all the services for the Perceptive Content Application server.
- Check for any other connections to the database and disconnect them.

Downtime steps

To execute the qualifying downtime schema changes necessary to take the INOW database schema to the specified target version, complete the following steps. The steps that are executed depends on the starting schema version and the specified target schema version.

IN_DB_UPGRADE_SP_DOWNTIME_STEPS

Note The IN_DB_UPGRADE_SP_DOWNTIME_STEPS procedure calls the IN_DB_UPGRADE_SP_UPTIME_STEPS procedure to ensure all prerequisite actions are completed.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_DOWNTIME_STEPS;
```

The following parameters are available to use.

- @MAX_DOP

Post downtime steps

Upon successful completion of the downtime steps, proceed with the following Perceptive Content Server related upgrade steps.

1. Perceptive Content Server related upgrade steps.
2. Full backup of the database.
3. Update database statistics.
4. Conduct standard user acceptance testing procedures.

Additional synchronization commands

For upgrades that start with schema version 7.5.0.0 and earlier, the following procedures can be executed as necessary to reset or remove the synchronization framework for the IN_DOC table. These commands are available between setup and downtime.

IN_DB_UPGRADE_SP_SYNC_RESET

To reset the synchronization framework between the IN_DOC table and the NEW_DOC table, complete the following step.

This will truncate the NEW_DOC table, drop any indexes on the table that were previously created during the uptime steps, reset the SYNC_SEQ_DOC sequence and truncate the SYNC_STAGE_DOC, SYNC_ERROR_DOC, SYNC_STATE_DOC, SYNC_HIST_DOC tables.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_RESET 'inuser.IN_DOC';
```

The following parameters are available to use.

- @TABLE_NAME (required)
- @MAX_DOP
- @FILLFACTOR

IN_DB_UPGRADE_SP_SYNC_REMOVE

To remove the synchronization framework between the IN_DOC table and the NEW_DOC table, complete the following step.

This will drop the following database objects which comprise the synchronization framework. SYNC_TRG_DOC trigger, SYNC_SEQ_DOC sequence, SYNC_STAGE_DOC table, SYNC_ERROR_DOC table, SYNC_STATE_DOC table, SYNC_HIST_DOC table, NEW_DOC table and indexes.

Note This also removes the record (for the IN_DOC table) from the IN_SYNC_CONTROL table.

- In SSMS or at the command prompt (sqlcmd), execute the following procedure.

```
EXEC inuser.IN_DB_UPGRADE_SP_SYNC_REMOVE 'inuser.IN_DOC';
```

The following parameter is required.

- @TABLE_NAME

Available parameters

When calling the various procedures you can specify one or more of the following parameters based on your preferences. Most procedures can be executed without specifying any additional parameters, in which case the default values contained in the IN_UPGRADE_CONTROL and IN_SYNC_CONTROL tables will be used.

Parameters

Note Not all parameters are available for every procedure.

Parameter	Description
-----------	-------------

@TARGET_SCHEMA_VERSION	<p>Specifies the schema version you want to upgrade the database to. Valid values are 7.5.0.0 and 7.7.0.0.</p> <p>The default value is 7.7.0.0.</p>
@FILEGROUP_NAME	<p>Specifies the name of the filegroup in which to create the primary objects.</p> <p>The default is either the current filegroup for the object or the default filegroup for the database.</p>
@MAX_MINUTES	<p>Specifies the number of minutes used to define an end time that a synchronization or upgrade task is allowed to start new operations.</p> <p>The default is 0 which is equal to unlimited.</p>
@ MAX_DOP	<p>Specifies the degree of parallelism for the session. Used during initial loading of the destination table and index creation.</p> <p>The default is 0.</p>
@ FILLFACTOR	<p>Specifies the percentage to fill data and index pages. Used during the creation of tables and indexes created during uptime and downtime operations.</p> <p>The default is 100.</p>
@DEFER_FK_CHECK	<p>Used to indicate whether the validation of foreign key constraints, to and from the new IN_DOC table, will be deferred and manually executed after the upgrade.</p> <p>The default is NO.</p> <p>The Foreign Key validation is the longest part of the downtime. Deferring that step is useful for facilitating the shortest possible downtime.</p> <p>You can defer the Foreign Key validation by specifying 'YES' ('Y') for the DEFER_FK_CHECK parameter when executing the downtime steps.</p> <p>If you defer the Foreign Key validation then the system displays the validation commands instead of executing them so you can manually execute them after the downtime steps.</p> <p>WARNING</p> <p>The Foreign Key check deferral delays the identification of any data consistency issues and could result in time consuming troubleshooting efforts after the upgrade.</p>
@BATCH_SIZE	<p>Used by the IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC procedure to define the batch size (number of rows) to populate the work queue (cursor) per batch.</p> <p>The default is 50,000 rows.</p>
@KEEP_SRC_TABLE	<p>Specifies whether to keep the original/source table (IN_DOC) upon completion of the downtime steps.</p> <p>The default is YES.</p>
@DEBUG	<p>Specifies whether to display additional logging for debugging purposes. Includes state changes, call stack and other details.</p>

	The default is NO.
--	--------------------

Examples

The following examples show some of the procedures with parameters specified during execution.

Specification of the parameters are not required during execution unless you want to override the default values. If you do not specify a parameter, then the default values are used.

```
EXECUTE inuser.IN_DB_UPGRADE_SP_UPGRADE_SETUP
  @TARGET_SCHEMA_VERSION = '7.7.0.0',
  @FILEGROUP_NAME = 'PRIMARY',
  @DEFAULT_MAX_MINUTES = 60,
  @DEFAULT_MAX_DOP = 4,
  @DEFAULT_FILLFACTOR = 90,
  @DEFAULT_BATCH_SIZE = 50000,
  @SYNC_DEFER_FK_CHECK = 'NO',
  @SYNC_KEEP_SRC_TABLE = 'YES';

EXECUTE inuser.IN_DB_UPGRADE_SP_UPTIME_STEPS
  @MAX_MINUTES = 60,
  @MAX_DOP = 4,
  @FILLFACTOR = 90;

EXECUTE inuser.IN_DB_UPGRADE_SP_SYNC_SETUP_IN_DOC
  @MAX_MINUTES = 60,
  @MAX_DOP = 0,
  @BATCH_SIZE = 50000,
  @DEFER_FK_CHECK = 'NO',
  @KEEP_SRC_TABLE = 'YES',
  @FILEGROUP_NAME = 'PRIMARY',
  @FILLFACTOR = 90;

EXECUTE inuser.IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC
  @MAX_MINUTES = 10,
  @BATCH_SIZE = 50000;

EXECUTE inuser.IN_DB_UPGRADE_SP_SYNC_REMOVE
  @TABLE_NAME = 'inuser.IN_DOC';

EXECUTE inuser.IN_DB_UPGRADE_SP_SYNC_RESET
  @TABLE_NAME = 'inuser.IN_DOC',
  @MAX_DOP = 4,
  @FILLFACTOR = 90;

EXECUTE inuser.IN_DB_UPGRADE_SP_CREATE_INDEXES_IN_DOC
  @FILEGROUP_NAME = 'SECONDARY',
  @MAX_MINUTES = 30,
  @MAX_DOP = 4,
  @FILLFACTOR = 90;

EXECUTE inuser.IN_DB_UPGRADE_SP_DOWNTIME_STEPS
  @MAX_DOP = 4;
```

View and update Upgrade parameter default values

You can manually update the `IN_UPGRADE_CONTROL` table to change the default values. To view the current parameter defaults, complete the following step. These default values are used by the various procedures that perform the upgrade steps unless otherwise specified during execution.

- In SSMS or at the command prompt (sqlcmd), execute the following command.

```
SELECT * FROM inuser.IN_UPGRADE_CONTROL;
```

Notes

The values shown below are the only values that should ever be modified in this table. Do not modify any other values.

The values used in the following example are not the defaults in some cases. See the parameter descriptions above for defaults.

```
UPDATE inuser.IN_UPGRADE_CONTROL SET
  DEFAULT_MAX_MINUTES = 60
, DEFAULT_MAX_DOP = 4
, DEFAULT_FILLFACTOR = 100;
```

View and update Synchronization parameter default values

You can manually update the `IN_SYNC_CONTROL` table to change the default values. To view the current parameter defaults, complete the following step. These default values are used by the `IN_DB_UPGRADE_SP_SYNC_TABLE_IN_DOC` procedure when synchronizing the new `IN_DOC` table (`NEW_DOC`) with the original `IN_DOC` table. Unless otherwise specified during execution the procedure will use the values specified in this table.

- In SSMS or at the command prompt (sqlcmd), execute the following command.

```
SELECT * FROM inuser.IN_SYNC_CONTROL;
```

Notes

The values shown below are the only values that should ever be modified in this table. Do not modify any other values.

The values used in the following example are not the defaults in some cases. See the parameter descriptions above for defaults.

```
UPDATE inuser.IN_SYNC_CONTROL SET
  DEFAULT_MAX_MINUTES = 30
, DEFAULT_MAX_DOP = 1
, DEFAULT_BATCH_SIZE = 50000
, DEFER_FK_CHECK = 'NO'
, KEEP_SRC_TABLE = 'YES';
```

Troubleshooting commands

- To view active session details, execute the following procedure.

```
EXEC inuser.IN_DB_UTIL_SP_SESSIONS_DISPLAY;
```

- To view call stack details, execute the following procedure.

```
EXEC inuser.IN_DB_UTIL_SP_CALLSTACK_PRINT;
```

- To view the contents of the global temporary table containing active session details and attributes, execute the following command. This table is queried by the IN_DB_UTIL_SP_SESSIONS_DISPLAY procedure, which should be used instead.

```
SELECT * FROM inuser.##IN_DB_UTIL_SESSION_ATTRIBUTES ORDER BY IDENTIFIER;
```

- To view the contents of the global temporary table containing call stack depth between stored procedures execute the following command. This table is queried by the IN_DB_UTIL_SP_CALLSTACK_PRINT procedure, which should be used instead.

```
SELECT * FROM inuser.##IN_DB_UTIL_CALLSTACK ORDER BY IDENTIFIER, DEPTH;
```

- To view the current state of synchronization, execute the following. This table is queried by the IN_DB_UPGRADE_SP_SYNC_GET_STATUS procedure, which can be used instead.

```
SELECT * FROM inuser.SYNC_STATE_DOC;
```

- To view the history of synchronization activity, execute the following command. This table is queried by the IN_DB_UPGRADE_SP_SYNC_GET_STATUS procedure, which can be used instead.

```
SELECT * FROM inuser.SYNC_HIST_DOC ORDER BY SYNC_DATETIME;
```

- To view the history of synchronization errors, execute the following command. This table is queried by the IN_DB_UPGRADE_SP_SYNC_GET_STATUS procedure, which can be used instead.

```
SELECT * FROM inuser.SYNC_ERROR_DOC ORDER BY ACTION_DATETIME;
```

Cleaning up after an early exit

You can use the following procedures to manually clear the call stack and session attribute tables as needed due to an unhandled exception or after manually cancelling an operation before completion.

Important Do not execute any of the following commands if any operations are still in progress as it will clear the call-stack and session attributes which will lead to undesired results and unhandled exceptions.

- Clear session and call stack for all identifiers (SYNC and UPGRADE)

```
EXEC inuser.IN_DB_UTIL_SP_CLEAR_IDENTIFIER 'ALL';
```

- Clear session and call stack for only the SYNC identifier

```
EXEC inuser.IN_DB_UTIL_SP_CLEAR_IDENTIFIER 'SYNC';
```

- Clear session and call stack for only the UPGRADE identifier

```
EXEC inuser.IN_DB_UTIL_SP_CLEAR_IDENTIFIER 'UPGRADE';
```

Debugging

You can use the @DEBUG parameter with most procedures if additional debugging is necessary.

This results in a very verbose execution that contains state change information and call stack details and other background call details to assist with debugging certain issues.

```
@DEBUG = 'YES'
```

Removing the Upgrade Package

You can use the following procedure to remove the upgrade package functions and procedures upon completion of the database upgrade.

```
EXEC inuser.IN_DB_UTIL_SP_DROP_UPGRADE_PROCS;
```