

# APS to Azure Synapse Analytics Migration Guide

- With GitHub [\(link\)](#) Open Source Tools

Prepared for APS to Azure Synapse Analytics Migration Practitioners

## Change Record

Date	Author	Version	Change Reference
March 2021	Gaiye 'Gail' Zhou	1.3	Initial version
June 2021	Aruna Dadi Andrey Mirskiy	1.4	Updates to reflect latest PowerShell toolkit version

## Table of contents

Migration Step by Step Guide.....	2
Environment and Workstation Preparations .....	2
1.1. Prepare APS Environments.....	2
1.2. Create Azure Subscription and Resources .....	3
1.3. Setup Workstation(s) or Laptop(s) .....	5
1.4. Prepare APS (T-SQL Scripts) .....	6
1.5. Prepare Azure Synapse Analytics (T-SQL Scripts) .....	6
1.6. Prepare Schema Mapping Matrix.....	6
Download and Setup the Open-Source Migration Tools.....	7
2.1. Download Migration Tools .....	7
2.2. Setup PowerShell Scripts Permissions .....	8
Implementation Steps and Output .....	10
3.1. Step 1: Obtain APS DDLs and DMLs.....	10
3.2. Step 2: Translate APS DDLs and DMLs .....	10
3.3. Step 3: Create Export and Import T-SQL Scripts .....	10
3.4. Step 4: Create T-SQL Scripts for Azure Synapse External Tables .....	10
3.5. Step 5: Deployments .....	11
3.6. Spot check Table and stored procedures and resolve issues.....	11
3.7. POC Results Verification Tests .....	11
Migration Exceptions Handling .....	12
4.1. Additional Code Conversion.....	12
4.2. Step 5: Deployments .....	12
Appendix T-SQL Scripts (Samples) .....	12

# Migration Step by Step Guide

## Environment and Workstation Preparations

Note: Please extract the zip file named [T-SQL-Scripts-Samples.zip](#) in (Appendix T-SQL Scripts (Samples)).

### 1.1. Prepare APS Environments

- **Set up Polybase in APS** – Please refer to [Setup\\_APS\\_Polybase\\_Sample.sql](#) for details.
- **Check APS Collation using T-SQL**

**SELECT CONVERT (varchar, SERVERPROPERTY('collation')) AS 'Server Collation';**

Sample Return Result: [Latin1\\_General\\_100\\_CI\\_AS\\_KS\\_WS](#)

### 1.2. Create Azure Subscription and Resources

- **Azure Subscription:** [Name here](#)
- **Azure Resource Region:** North Europe
- **Azure Storage (Blob Storage) Name:** [Blob Storage Account Name](#)
  - Storage Settings for Polybase
    1. **Performance:** [Standard](#)
    2. **Security transfer required:** [Enabled](#)
    3. **Access tier** (default): [Hot](#)
    4. **Replication:** [Locally redundant storage \(LRS\)](#)
  - Connectivity: [Public endpoint \(all networks\)](#)
  - Security
    1. Security transfer required: [Enabled](#)
    2. Blob soft deletion: [Disabled](#)
    3. Data Lake Storage Gen2 (Hierarchical namespace): [Disabled](#)
  - Create containers with below suggested names:
    1. [pocmigrations](#): Use this container for APS->ASA migration POC.
    2. [pocdelta](#): Use this container for data ingestion process (Sources->ASA) POC.
    3. [migrations](#): Use this container for production migrations.
- **Azure Storage General Purpose V2 (Blob Storage)**

## Configuration

«

Save Discard

Account kind

StorageV2 (general purpose v2)

Performance ⓘ

Standard Premium

Secure transfer required \* ⓘ

Disabled Enabled

Access tier (default) ⓘ

Cool Hot

Replication ⓘ

Locally-redundant storage (LRS)

Large file shares ⓘ

Disabled Enabled

Identity-based Directory Service for Azure File Authentication ⓘ

None

Data Lake Storage Gen2

Hierarchical namespace ⓘ

Disabled Enabled

+ Container

Change access level

Refresh

Delete

Search containers by prefix

Name

☐ migrations

☐ pocdelta

☐ pocmigrations

○ Azure Synapse Analytics: Server name and db name

- Create Azure Synapse Analytics (formerly Azure SQL DW). **Collation Choice examples:**
  1. Use Azure Synapse default collation which is "SQL\_Latin1\_General\_CP1\_CI\_AS". Then all the APS creates table statements need to replace original APS collation with blank. For example, Replace "Latin1\_General\_100\_CI\_AS\_KS\_WS" with blank string " ". **If you don't replace them with empty string, you may encounter problems. Please do this.**
  2. Create Azure Synapse Analytics with default collation unless otherwise specified by the customer. This applies to most of the situations.
  3. Create Azure Synapse Analytics (formerly Azure SQL DW) with the **same collation** as current APS collation if that is appropriate / required.
- In Azure Synapse Analytics, the Collation cannot be changed after data warehouse creation. The default collation is SQL\_Latin1\_General\_CP1\_CI\_AS. For more details, please read this online [documentation page](#).
- In ASA, Check Azure SQ DW Collation by the following T-SQL  

```
SELECT CONVERT (varchar, SERVERPROPERTY('collation')) AS 'Server Collation';
```

Sample Return Result:

ADW: SQL\_Latin1\_General\_CP1\_CI\_AS

APS: Latin1\_General\_100\_CI\_AS\_KS\_WS

- Check Azure SQ DW DB Collation by the following T-SQL  

```
SELECT DATABASEPROPERTYEX(DB_NAME(), 'Collation') AS Collation;
```

Sample Return Result: SQL\_Latin1\_General\_CP1\_CI\_AS

- List all supported collations in Azure SQ DW by the following T-SQL (connect to master)  

```
SELECT * FROM sys.fn_helpcollations();
```

#### ○ **Azure Active Directory & Security Setup**

- Onboarding project team members to access customer network (Laptop issued by customer in some cases)
- Set up access for project **team members to access Azure Resources** (Customer Azure Security Admin performs this after all Azure Resources have been created).

## 1.3. Setup Workstation(s) or Laptop(s)

- **Network connectivity:** Connectivity to Internet, APS, and Customer Azure Resources.
- **Customer team communications:** Connectivity to Customer internal communications: Email, team sites, etc.
- **Data Tools Setup**
  1. Visual Studio 2017 and SSDT to run APS T-SQL Scripts (This is required for APS access)
  2. SSMS: Latest Version to run ASA T-SQL Scripts (optional but recommended)
  3. Visual Studio Code or other user preferred IDEs to run PowerShell Scripts and Python Programs

## 1.4. Prepare APS (T-SQL Scripts)

- **T-SQL Scripts:** [Setup\\_APS\\_Polybase.sql](#) (Run by PFE or APS Admin)
- **T-SQL Scripts:** [Setup\\_APS\\_Migration\\_User.sql](#) (Run by PFE or APS Admin)
- **T-SQL Scripts:** [Setup\\_APS\\_Export.sql](#) (Run by Migration User)

Sample T-SQL Scripts are provided in Section 0, "Migration Exceptions Handling". For APS roles and user permission, please refer to APS Document (click [here](#)).

## 1.5. Prepare Azure Synapse Analytics (T-SQL Scripts)

- **T-SQL Scripts:** [Setup\\_ASA\\_Migration\\_User.sql](#) (Run by PFE or ASA Admin)
- **T-SQL Scripts:** [Setup\\_ASA\\_Import.sql](#) (Run by Migration User)

Sample T-SQL Scripts are provided in Section 0, "Migration Exceptions Handling".

## 1.6. Prepare Schema Mapping Matrix

- **Prepare Schema Mapping File:** [schemas.csv](#)
- **Below is a sample schema mapping file** (contents of schemas.csv):

ApsDbName	ApsSchema	SQLDWSchema
adventureworks_stage	dbo	stage_dbo
adventureworks_edw	dbo	edw_dbo
adventureworks_edw	hr	edw_hr
adventureworks_datamarts	hr	dm_hr
adventureworks_datamarts	finance	dm_finance

# Download and Setup the Open-Source Migration Tools

## 2.1. Download Migration Tools

The [Github](#) contains overview of the migration process and very detailed step by step instructions for each of the five steps. This same process and scripts have been successfully utilized in production migration projects.

Download from [this link](#) and save the zipped file "AzureSynapseScriptsAndAccelerators-main.zip" to workstation or laptop.

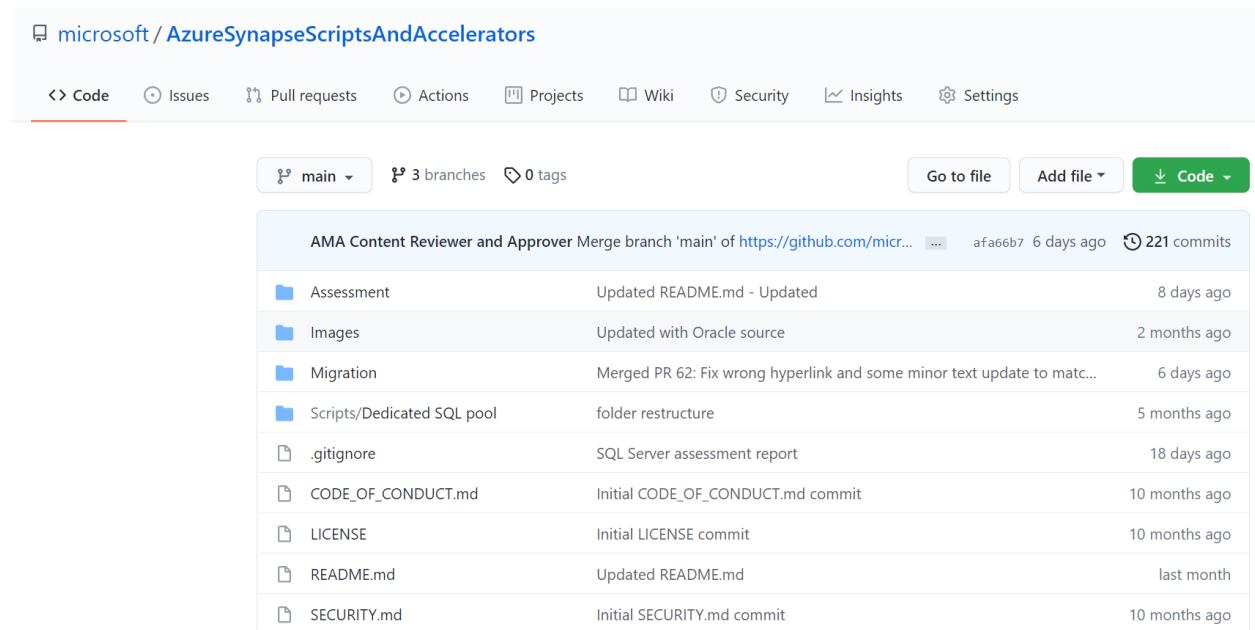


Figure 1 Download Migration Tools from Github

Unzipped (Extract All) the downloaded into a designated directory with the default folder name: "AzureSynapseScriptsAndAccelerators-main". This folder contains all the assessment & migration utilities.

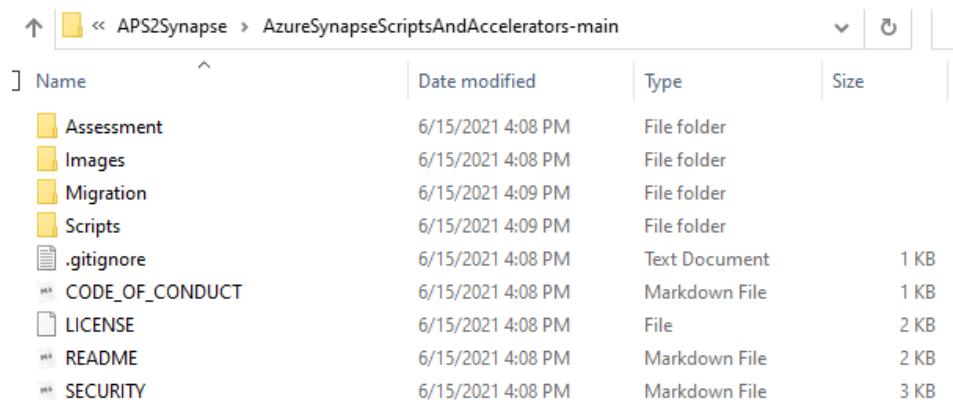
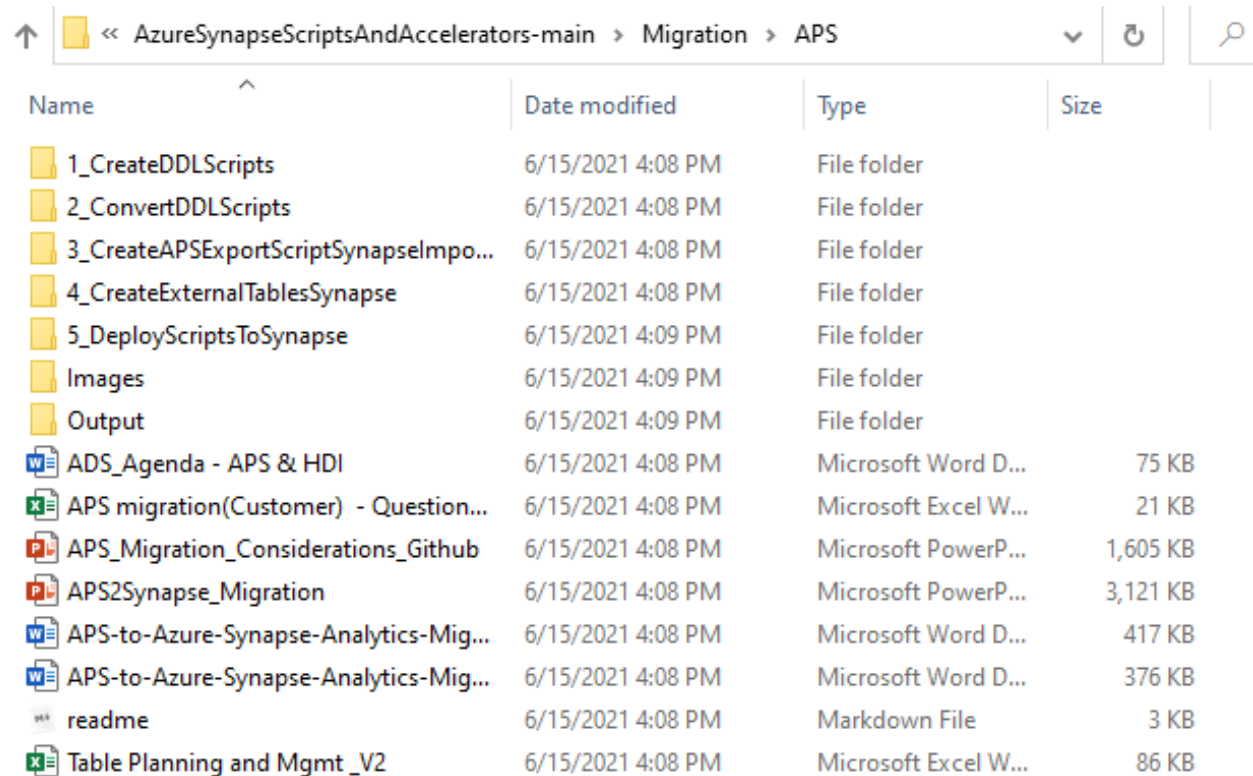


Figure 2 Migration Tools File Structure

You can move this “AzureSynapseScriptsAndAccelerators-main” to any location (directory) of your choice. Migration files are illustrated in Figure 3 below. From there, many of the instructions in [Github](#) can be directly applied. In addition, the configuration files (.csv) will need minimal modifications. We also have job-aid powershell scripts to help automate the configuration file generation process (for example, step 3: [Generate\\_Step3\\_ConfigFile.ps1](#)).



Name	Date modified	Type	Size
1_CreateDDLScripts	6/15/2021 4:08 PM	File folder	
2_ConvertDDLScripts	6/15/2021 4:08 PM	File folder	
3_CreateAPSEXPExportScriptSynapseImpo...	6/15/2021 4:08 PM	File folder	
4_CreateExternalTablesSynapse	6/15/2021 4:08 PM	File folder	
5_DeployScriptsToSynapse	6/15/2021 4:09 PM	File folder	
Images	6/15/2021 4:09 PM	File folder	
Output	6/15/2021 4:09 PM	File folder	
ADS_Agenda - APS & HDI	6/15/2021 4:08 PM	Microsoft Word D...	75 KB
APS migration(Customer) - Question...	6/15/2021 4:08 PM	Microsoft Excel W...	21 KB
APS_Migration_Considerations_Github	6/15/2021 4:08 PM	Microsoft PowerP...	1,605 KB
APS2Synapse_Migration	6/15/2021 4:08 PM	Microsoft PowerP...	3,121 KB
APS-to-Azure-Synapse-Analytics-Mig...	6/15/2021 4:08 PM	Microsoft Word D...	417 KB
APS-to-Azure-Synapse-Analytics-Mig...	6/15/2021 4:08 PM	Microsoft Word D...	376 KB
readme	6/15/2021 4:08 PM	Markdown File	3 KB
Table Planning and Mgmt_V2	6/15/2021 4:08 PM	Microsoft Excel W...	86 KB

Figure 3 Recommended Location for Migration Tools

## 2.2. Setup PowerShell Scripts Permissions

### 2.2.1. Option 1

When you start PowerShell on a computer for the first time, the **Restricted** execution policy (the default) is likely to be in effect. The **Restricted** policy does not permit any scripts to run.

To find the effective execution policy on your computer (PowerShell): [Get-ExecutionPolicy -list](#)

To run unsigned scripts that you write on your local computer and signed scripts from other users, start PowerShell with the Run as Administrator option and then use the following command to change the execution policy on the computer to **RemoteSigned**:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope LocalMachine
```



To run an unsigned script, use the Unblock-File cmdlet ([Unblock-File -Path .\fileName.ps1](#)). For more information, please refer to this PowerShell documentation page (link [here](#)).

If you are using Visual Studio Code to run PowerShell Scripts, you need to run the VSC as administrator in order to change Policy.

### 2.2.2. Option 2

To run unsigned scripts that you write on your local computer and signed scripts from other users, start PowerShell with

```
Set-ExecutionPolicy Unrestricted
```

You can also pin Powershell IDE to taskbar, then Shift + right-click to Run with Elevated Rights and then run below script:

```
Set-ExecutionPolicy Unrestricted
```

### 2.2.3. Install PowerShell Modules

Install sqlserver and ImportExcel modules for Powershell

From pinned Powershell Shift Right-Click Run with Elevated Rights

```
Install-Module sqlserver  
Install-Module -Name ImportExcel
```

## Implementation Steps and Output

This section describes the implementation steps and output of each step.

All the migration scripts with customer configurations to deliver the output will be delivered to customer.

For more detailed instructions for each of the five steps, please visit [GitHub](#).

### 3.1. Step 1: Obtain APS DDLs and DMLs

- **Step 1A:** Prepare Configuration Files to run migration tools step 1, create APS scripts
- **Step 1B:** Run PowerShell script **CreateDDLScriptsDriver.ps1**.
- **Output:** T-SQL scripts for APS objects (DDLs and DMLs)

For more detailed instructions for this step, please this GitHub Page: [Step 1 Instructions](#).

### 3.2. Step 2: Translate APS DDLs and DMLs

- **Step 2A:** Prepare configuration files (Schema mapping csv file)
- **Step 2B:** Translate APS DDLs and DMLs into Azure Synapse Analytics T-SQL Scripts.
- **Input:** Output from Step 1
- **Output:** T-SQL Scripts for Azure Synapse Analytics

For more detailed instructions for this step, please this GitHub Page: [Step 2 Instructions](#).

### 3.3. Step 3: Create Export and Import T-SQL Scripts

- **Step 3A:** Prepare the configuration CSV files
- **Step 3B:** Create T-SQL Scripts for Exporting APS tables and Importing the same into Azure Synapse Analytics
- **Input:** Output from Step 2
- **Output:** Export, Copy Into and Import T-SQL Scripts

For more detailed instructions for this step, please this GitHub Page: [Step 3 Instructions](#).

### 3.4. Step 4: Create T-SQL Scripts for Azure Synapse External Tables

- **Step 4A:** Create configuration files
- **Step 4B:** Create T-SQL Scripts for Azure Synapse Analytics External Tables (which point to location of data stored in Azure Storage)
- **Input:** Output from Step 2
- **Output:** T-SQL Scripts for Azure Synapse Analytics External Tables (which point to location of data stored in Azure Storage)

For more detailed instructions for this step, please this GitHub Page: [Step 4 Instructions](#).

## 3.5. Step 5: Deployments

This step performs tasks using outputs from previous steps 1-5. The overall output of these deployment steps: DDLs and DMLs created in ASA. External Tables are created in ASA. Tables are exported from APS to Blob Storage, and then imported from Blob Storage to ASA.

The tasks can be broken down into below groups:

- **Step 5A:** Run T-SQL Scripts to create Tables, Views, and Stored Procedures in Azure Synapse Analytics
- **Input:** Output from Step 2
- **Results of 5A:** Tables, Views, and Stored Procedures created in Azure Synapse Analytics
  
- **Step 5B:** Run T-SQL Scripts to create external tables in Azure Synapse Analytics
- **Input:** Output from Step 4
- **Results of 5B:** External Tables created in Azure Synapse Analytics
  
- **Step 5C:** Run APS Export Scripts
- **Input:** Output from Step 3
- **Results of 5C:** APS data exported to Azure Storage
  
- **Step 5D:** Run Azure Synapse Analytics Import Scripts
- **Input:** Output from Step 3
- **Results of 5D:** Data is imported into Azure Synapse Analytics

For more detailed instructions for this step, please this GitHub Page: [Step 5 Instructions](#).

## 3.6. Spot check Table and stored procedures and resolve issues

- **Spot Check 2 Tables:** Check # of records, check contents, check distributions, partitions.
- **Spot Check 2 Views:** Compile and execute T-SQL scripts from the views
- **Spot Check 2 Stored Procedures:** Compile and run the stored procedures with default values

## 3.7. POC Results Verification Tests

- **POC Results Verification Tests:** Customer team will test Azure Synapse Analytics with T-SQL Scripts to verify the POC results. Customer team will document issues/errors and feedback to CSE team.

## Migration Exceptions Handling

This section provides tips on exceptions handlings. Not all issues may arise in your migration project. If they do, these tips will provide additional help.

### 4.1. Additional Code Conversion

If your code is imbedded in configuration tables, the m

Table: EtlJobsAction

RecordID	Condition	CodeToExecute
1	Condition_A	... [ApsDbName].sys.tables ...
2	Condition_B	... [APsDbName].sys.views ...
8	Condition_X	... SET @Msg = 'Dropped the table ' + @schema + '.' + @tablename + '_new if existed' ...

The code containing "... [ApsDbName].sys.tables ..." needs to be replaced with "... sys.tables ...". The code containing "... [ApsDbName].sys.views ..." needs to be replaced with "... sys.views ...". For a complete list of examples like these, please review to

### 4.2. Step 5: Deployments

## Appendix T-SQL Scripts (Samples)

This section provides sample T-SQL Scripts that are used to set up APS and Azure Synapse Analytics environments. For security reasons, things like server name, database name, azure storage keys are made up values (samples). Users of these scripts will need to replace them with actual values to run these T-SQL Scripts.

All the T-SQL Scripts can be found in this attached zipped file: T-SQL-Scripts-Samples.zip.



T-SQL-Scripts-Samples.zip