# AWS Modernization Tools for Microsoft Workloads

## REFERENCE GUIDE

This guide is intended for architects and developers who are exploring modernizing Microsoft Workloads (.NET applications and SQL Server databases) to AWS. It provides guidance on the AWS modernization tools and recommended approaches to help you accelerate your journey to cloud.

The flow diagram showcases how the AWS modernization tools can help to accelerate modernization at every stage of the modernization journey. These tools can be used in conjunction with each other or individually based on the requirements.

### CONTENTS



https://aws.amazon.com/windows/modernization/

https://aws.amazon.com/dotnet

https://twitter.com/dotnetonaws

# AWS Modernization Tools for Microsoft Workloads

## Discovery

Microsoft Workloads (.NET Applications and SQL Server Databases)

For calculating the cost of running modernized Microsoft workloads

AWS Modernization Calculator for Microsoft Workloads

Migration Hub Strategy Recommendations

Migration Hub Strategy Recommendations discovers applications & databases and suggests different modernization paths and corresponding tooling

Web Applications

ASP.NET MVC Application

ASP.NET Web Forms Application

WCF Service

Windows Service

Microsoft SQL Server Database

Refactor to micro-services and .NET6+

No

Refactor to .NET6+

No

Replatform

Yes

Yes

Yes

Refactor

No

Replatform

Yes

Yes

## Assessment

Visual recommendations for monolithic to microservice(s) along with porting changes

Analyze .NET Framework source code for .NET 6+ compatibility

No

Analyze code in Visual Studio

Yes

For analyzing SQL scripts for compatibility with Babelfish for PostgreSQL

For analyzing schema conversion from SQL Server to MySQL, PostgreSQL or Amazon Redshift

AWS Microservice Extractor for .NET

Porting Assistant for .NET (Standalone tool)

AWS Toolkit for .NET Refactoring

Babelfish Compass

AWS Schema Conversion Tool

## Modernization

Microservice(s) Extraction and optional porting to .NET6+

Assistive porting of .NET Framework application to .NET 6+ & validating code changes by test deploying to AWS from Visual Studio

Applications picked from IIS and Windows services list, containerized and image is pushed to Amazon ECR

For running SQL Server applications targeting Aurora PostgreSQL with little to no code changes

For migration of SQL database and analytics workloads to AWS

For re-platforming from on-prem SQL Server to Amazon RDS SQL Server

AWS Microservice Extractor for .NET

AWS Toolkit for .NET Refactoring

AWS App2Container

Babelfish for Aurora PostgreSQL

AWS Database Migration Service (DMS)

## Final state

Apps ported to .NET 6+

.NET 6+ apps

ASP.NET MVC App on .NET 6+

ASP.NET Blazor App on .NET 6+

WCF Core App on .NET 6+

Windows Service on .NET 6+

Babelfish converting calls between application and target Aurora PostgreSQL

Babelfish for Aurora PostgreSQL

Amazon Aurora PostgreSQL

AWS DMS migrating data to Amazon RDS SQL

Amazon RDS for SQL Server Database

Monolithic application split into microservices and ported to latest framework

.NET 6+ app with microservices

ASP.NET MVC App with micro-services

ASP.NET Blazor App with micro-services

WCF Core App with micro-services

Applications are containerized (Windows containers) and hosted on ECS/EKS

Windows Containers on Amazon ECS/EKS

ASP.NET MVC Application

ASP.NET Web Forms Application

WCF Service

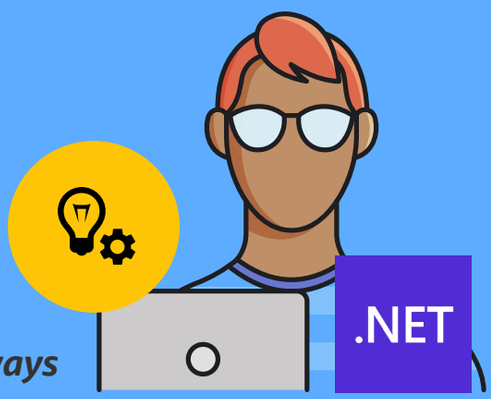
Windows Service

AWS DMS can migrate data to Aurora MySQL or Aurora PostgreSQL or Amazon Redshift

Amazon Aurora MySQL/ PostgreSQL or Amazon Redshift

# AWS Migration Hub Strategy Recommendations

*Easily identify migration & modernization pathways*




## What

Part of AWS Migration Hub, Strategy Recommendations (MHSR) analyzes your servers, application source-code, and databases to provide prescriptive modernization recommendations.

## Why

- ✓ Identify the best-suited option
- ✓ Portfolio-wide recommendations
- ✓ Understand how to migrate & modernize
- ✓ Find the low-hanging fruit
- ✓ Make informed decisions

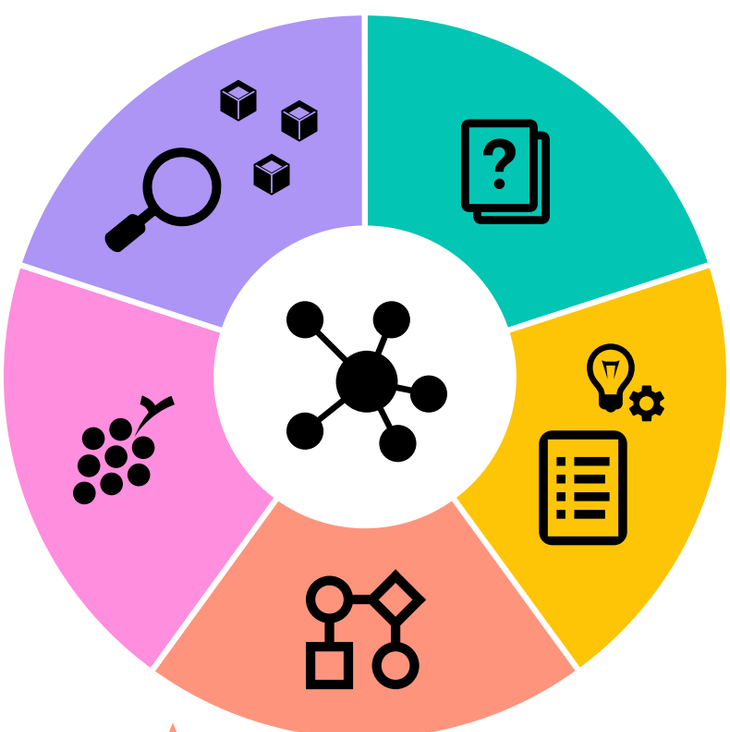
## How

1

**Configure Data Sources**
Add application data through an agentless app data collector or app data import.

2

**Add Preferences**
Answer a few questions about your business preferences, such as goals.



**Find the Low-Hanging Fruit**
MHSR identifies apps you can easily modernize as you migrate, with minimal effort, using existing AWS and partner tools.

3

**Receive Recommendations**
Receive AWS recommendations for the migration and modernization of your applications.

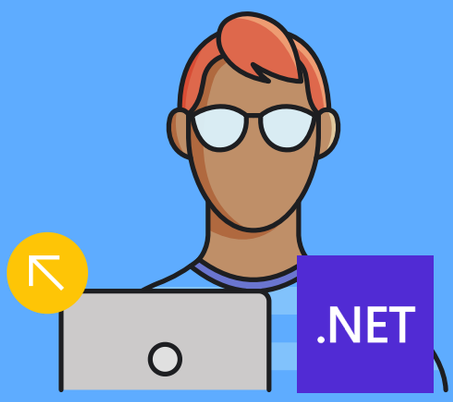
**Understand How to Migrate and/or Modernize**
MHSR helps you understand the effort and the tools you can use to automate.

**Learn more:**
https://aws.amazon.com/migration-hub/features

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# Porting Assistant for .NET

*Insight and assistance for porting .NET Framework applications to modern .NET*




## What

Porting Assistant for .NET (PA) is an analysis tool that scans .NET Framework applications and generates a modern .NET compatibility assessment, helping you port your applications to Linux faster.

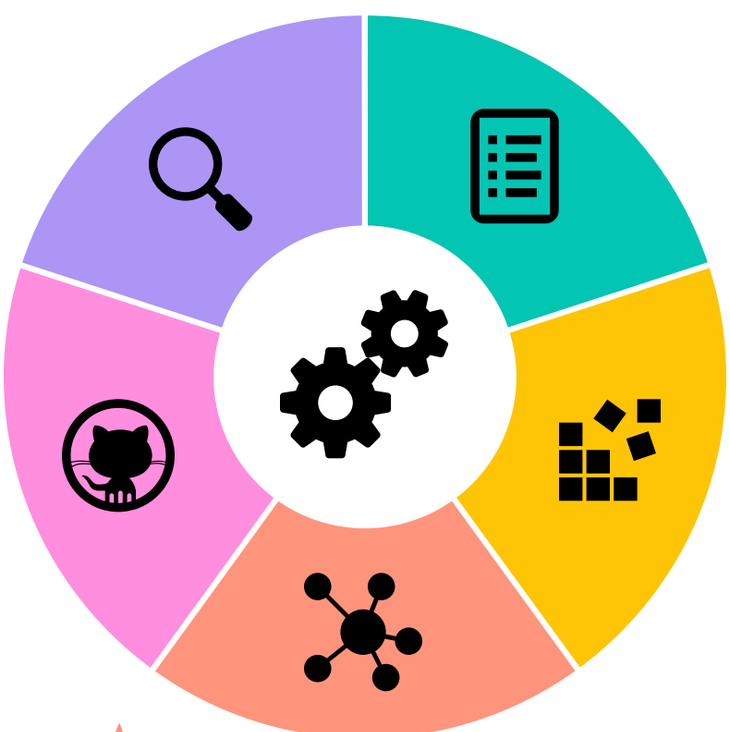
## Why

Porting .NET Framework applications to modern .NET helps customers take advantage of the performance, cost savings, and robust ecosystem of Linux. However, porting can be a significant manual effort. Application owners need to identify dependencies and resolve incompatible APIs.

## How

1

**Scan**
PA scans .NET Fx apps to find APIs & NuGet packages incompatible with .NET 6+.



2

**Assessment Report**
PA generates compatibility report and suggests available replacements.

3

**Assisted Porting**
PA updates packages and changes project reference files for you to start porting.

**Public Dataset**
The dataset PA uses is publicly available on GitHub and includes data AWS has curated as well as data from other public sources.

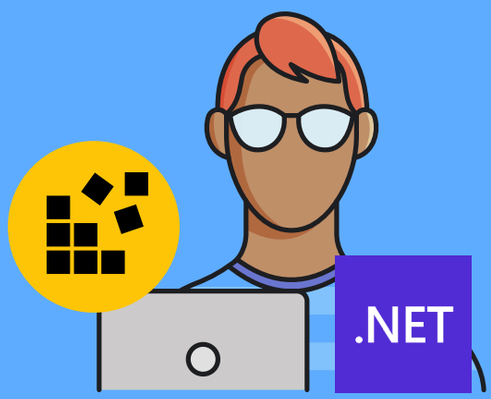
**Dependency Visualization**
PA GUI visualizes dependencies within a solution file, helping assess the impact of changes.

**Learn more:**
https://aws.amazon.com/porting-assistant-dotnet

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# AWS Microservice Extractor for .NET

*Simplify refactoring .NET applications*



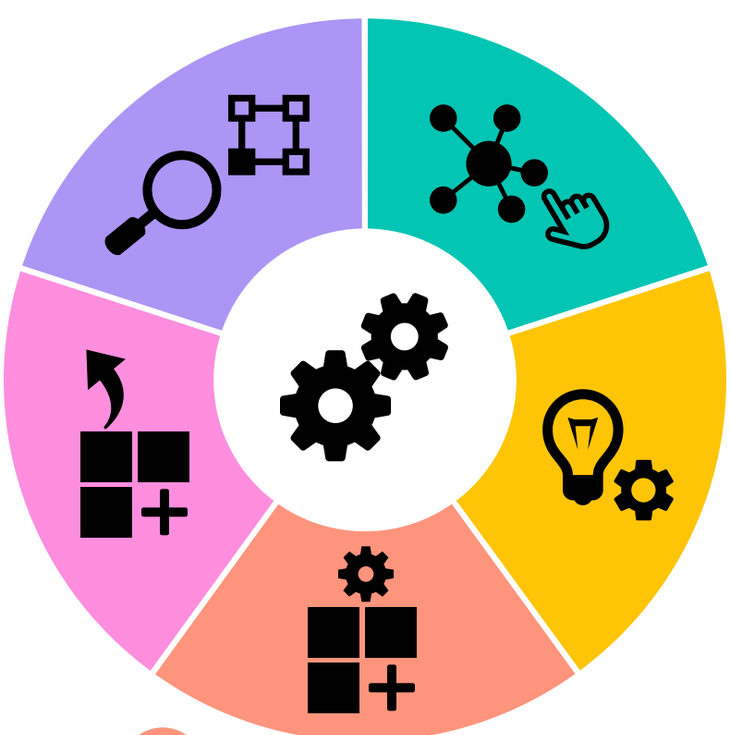

## What

AWS Microservice Extractor for .NET (MSE) simplifies the process of refactoring older monolithic apps into small code projects to build a microservices-based architecture.

## Why

- ✓ Faster app component identification
- ✓ Auto refactoring recommendations
- ✓ Assisted monolith refactoring
- ✓ Porting Assistant for .NET integration
- ✓ Available for use at no cost

## How



1

**Onboard Application**
Provide access to buildable source code. Optionally run the tool's profiler to collect runtime metrics.

2

**Launch Visualization**
Visually group and label parts of the application to create as independent services.

3

**Automated Recommendations**
MSE identifies common extract-ion candidates using heuristics and AI. You get a guided experience to refactor legacy applications.

4

**Refactor & Extract**
Refactor code by isolating business domains and removing dependencies. Extract groups as separate code repositories.

5

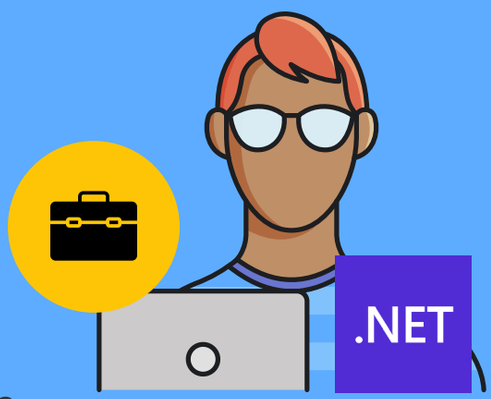
**Build & Deploy**
Manually refactor, build, and deploy the extracted services as well as the modified code.

**Learn more:**
https://aws.amazon.com/microservice-extractor

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# AWS Toolkit for .NET Refactoring

*Expedite .NET modernization in Visual Studio*




## What

AWS Toolkit for .NET Refactoring (TR) is a Visual Studio extension that helps developers refactor legacy .NET applications to cloud-based alternatives on AWS.

## Why

- ✓ Modernization recommendations
- ✓ Identifies IIS and AD dependencies
- ✓ Modifies code for Linux compatibility
- ✓ Validates refactored app on AWS
- ✓ All without leaving Visual Studio IDE

## How

1

**Assessment**
TR scans your .NET application and identifies refactoring pathways and configurations for Linux.

**One Tool to Learn**
TR eliminates the burden of learning multiple tools or interfaces. It removes the guesswork in app refactoring.

2

**Porting**
TR assists with code modifications to kickstart refactoring and enable Linux compatibility.

3

**Testing on AWS Environments**
Validate changes by running on AWS directly from Visual Studio.

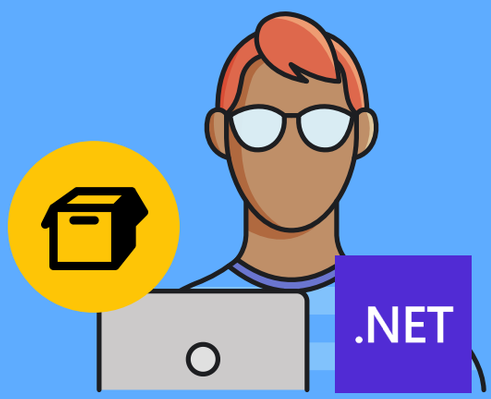
**Assisted Updates**
TR assists with changes to project reference files and Windows dependency configurations to reduce manual effort.

**Learn more:**
https://aws.amazon.com/visual-studio-net/

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# AWS App2Container

***Containerize and migrate .NET web applications***




## What

AWS App2Container (A2C) is a command line tool for migrating and modernizing .NET web applications running in IIS on Windows into container format. It also supports Java applications.

## Why

- ✓ Save on infrastructure & training costs
- ✓ Accelerate modernization
- ✓ Automated application analysis
- ✓ Auto-generated container images
- ✓ Containerize without code changes
- ✓ Built-in automation pipeline integration

## How

1

**Discover & Analyze**
Create application inventory of your ASP.NET applications and analyze runtime dependencies.

**Automated App Provisioning**
A2C generates CloudFormation templates that configure required infrastructure.

2

**Extract & Containerize**
Extract an application with dependencies and create a Docker image.

3

**Create Deployment Artifacts**
Generate the Amazon ECS task or Kubernetes pod definitions, and create CI/CD pipelines.

4

**Deploy to AWS**
Store the image in Amazon ECR, and deploy seamlessly to Amazon ECS or Amazon EKS.

**Learn more:**
https://aws.amazon.com/app2container

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# AWS Modernization Calculator for Microsoft Workloads

## What

Modernization Calculator for Microsoft Workloads (MCMW) helps you estimate the cost of modernizing your Microsoft workloads to a new architecture, using open-source software and cloud-native services on AWS.

## Why

- ✓ Estimate your modernization costs
- ✓ Assess benefits of cloud technologies
- ✓ Reduce total cost of ownership (TCO)
- ✓ Eliminate software license management
- ✓ No AWS expertise required
- ✓ No AWS account required

## How

1

**Identify current architecture**
Provide details about your application's current architecture and where it is deployed.

2

**Select an architecture size**
Provide current architecture characteristics to estimate the approximate size of the modernized architecture.

3

**Choose modern architecture**
Choose a recommended modernized architecture pattern for your application.

4

**Edit Service Configuration**
Review the estimated cost and adjust the recommended AWS services and settings.

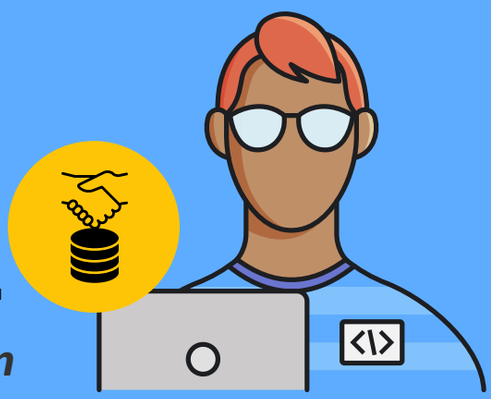
**An intuitive tool**
You can use MCMW without a detailed knowledge of AWS services.

**Learn more:**
https://modernization.calculator.aws/microsoft/estimate

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# Babelfish for Aurora PostgreSQL

***Run Microsoft SQL Server applications on PostgreSQL with little to no code change***



## What

Babelfish for Aurora PostgreSQL enables Amazon Aurora PostgreSQL-Compatible Edition to understand commands from applications written for Microsoft SQL Server.

## Why

- ✓ Move to an open source database
- ✓ Migrate SQL Server apps to Aurora
- ✓ Use same communication protocol
- ✓ Fewer code changes needed
- ✓ Accelerated, cost-effective migration
- ✓ Built-in capability, no additional cost

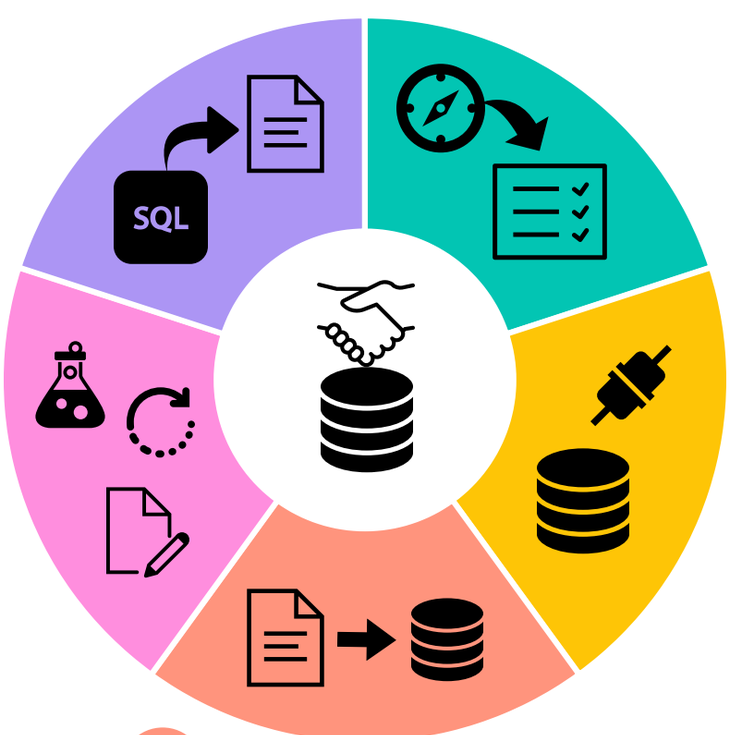
## How

**1**

**Generate DDL & Capture TSQL Code**
Generate DDL and run SQL Profiler or Extended Events trace on SQL database

**2**

**Assess & Update**
Run the Babelfish Compass tool to identify issues. Review the assessment report and rewrite or remove unsupported SQL features.




**3**

**Create Cluster & Connect**
Create the Babelfish cluster and connect to the Babelfish database.

**4**

**Run Updated DDL**
Run the updated DDL against the Babelfish database.

**5**

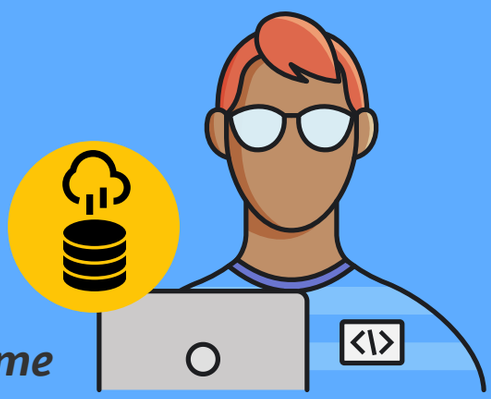
**Test & Iterate**
Test and iterate until the migrated application's functionality is correct.

**Learn more:**
https://aws.amazon.com/rds/aurora/babelfish/

**.NET on AWS Developer Hub**
https://aws.amazon.com/dotnet

# AWS Data Migration Service

*Migrate databases with minimal downtime*



## What

AWS Database Migration Service (AWS DMS) is a managed migration and replication service that helps move your database and analytics workloads to AWS quickly, securely, and with minimal downtime and zero data loss.

## Why

✓ Automated migration
✓ Maintain HA and minimal downtime
✓ Migrate to same or different DBMS
✓ Migrate securely at low cost

## How

1

**Discover & Assess**
DMS fleet advisor automatically inventories and assesses your server fleets.

2

**Convert**
DMS Schema Conversion converts the source scheme and code to match the target database.


**Simple to Use**
DMS is simple to use. There are no drivers or apps to install, and source database changes are not required in most cases.

3

**Migrate**
DMS creates a task to connect source and target databases and initiates migration.

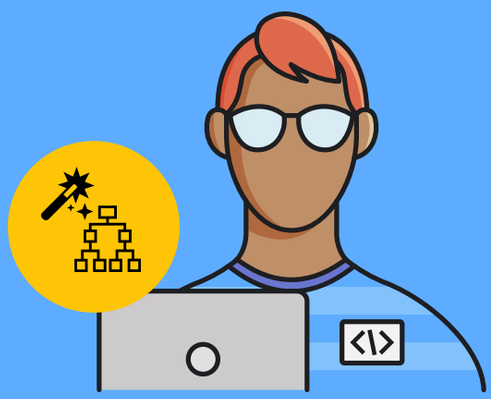
**Minimal Downtime**
Migrate your database to AWS with virtually no downtime. The source database is fully operational during the migration process.

**Learn more:**
https://aws.amazon.com/dms/

**AWS Cloud Databases**
https://aws.amazon.com/products/databases/

# AWS Schema Conversion Tool

*Convert database schemas from one database engine to another*



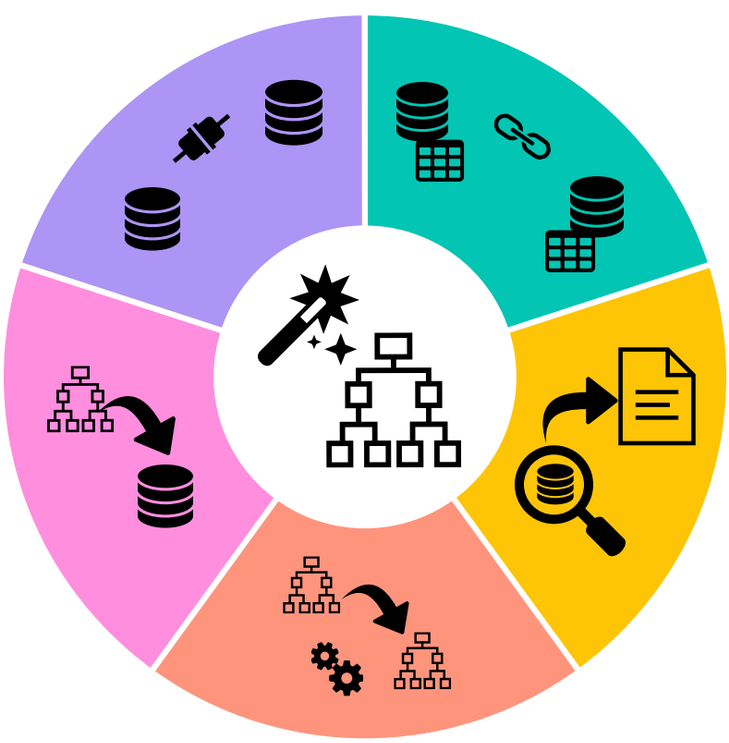
## What

AWS Schema Conversion Tool (SCT) simplifies database migrations by automating schema analysis, recommendations, and conversion at scale.

## Why

- ✓ Automated schema analysis
- ✓ Assessment identifies action items
- ✓ Converts views, SPs, & functions
- ✓ Converts embedded SQL in code

## How



**1**

**Create & Connect**
Create an SCT project, and connect to your source and target databases.

**2**

**Create Mapping Rules**
Create one or more schema mapping rules describing source-target pairs.

**3**

**Assess & Review**
Run and review the Database Migration Assessment Report to identify action items and estimate level of effort.

**4**

**Convert Schemas & Address Action Items**
Convert source database schemas. Respond to assessment action items by modifying schemas.

**5**

**Apply Schema Conversion**
Apply the converted schema to your target database.

**Learn more:**
https://aws.amazon.com/dms/schema-conversion-tool/

**AWS Cloud Databases**
https://aws.amazon.com/products/databases/

# Resources

.NET on AWS Developer Hub

## Technical Guides

Monolithic to Microservice Journey for .NET Applications Technical Guide

Modernize .NET Applications with Linux Containers Technical Guide

Replatform .NET Applications with Windows Containers Technical Guide

Microsoft SQL Server 2019 to Amazon Aurora PostgreSQL Migration Playbook

## Hands-on Workshops

AWS Microservice Extractor for .NET Workshop for ASP.NET MVC app

AWS Microservice Extractor for .NET Workshop for ASP.NET WebForms app

Migration Hub Strategy Recommendations workshop

Porting Assistant for .NET Workshop

App2Container DotNet Modernization Lab

Babelfish Immersion Day

Modernize .NET application from A to Z workshop

AWS Database Migration Workshop