Praveen Madupu +91 98661 30093
Sr SQL Server DBA
praveensqldba12@gmail.com

Here are the **options to migrate On-Premise SQL Server to AWS**, based on **target AWS service**, **migration strategy**, and **downtime tolerance**:

✅ 1. Migrate to Amazon RDS for SQL Server (Managed Service - PaaS)

Amazon RDS for SQL Server is a **managed relational database service** where AWS handles patching, backups, and monitoring.

🔹 Migration Methods:

1.1 AWS Database Migration Service (DMS)

- Supports **online (minimal downtime)** and **offline (one-time)** migrations
- Schema conversion with **AWS Schema Conversion Tool (SCT)**
- Handles **ongoing replication**
- Best for: **production workloads**, **heterogeneous or homogenous migration**

1.2 Native Backup and Restore to RDS

- Take `.bak` file from on-prem SQL Server
- Upload to **Amazon S3**
- Use **RDS stored procedures** to restore from S3
    - `rds_restore_database`
- Limitations: Must be **SQL Server 2008 R2 or later**
- Moderate downtime required

1.3 BACPAC Export/Import to RDS

- Export schema + data as a **.bacpac file**
- Use **SSMS** or **SQLPackage** to import into RDS
- Downtime needed; not suitable for large databases

Praveen Madupu +91 98661 30093
Sr SQL Server DBA
praveensqldba12@gmail.com

1.4 AWS Snowball for Large Data

- Ship encrypted backup using **Snowball device** to AWS
- Import data to RDS after delivery
- Best for **large datasets with limited internet bandwidth**

✅ 2. Migrate to SQL Server on Amazon EC2 (Self-Managed - IaaS)

SQL Server runs on an **EC2 instance** (virtual machine); you control everything.

🔹 Migration Methods:

2.1 Backup and Restore

- Take `.bak` file from on-prem
- Upload via **S3**, **AWS DataSync**, or **direct copy**
- Restore using **SSMS** on EC2 instance

2.2 Detach and Attach

- Detach `.mdf` and `.ldf` files
- Copy and attach on EC2
- Simple, but needs downtime

2.3 Log Shipping

- Setup log shipping from on-prem to EC2
- Cutover after sync for **low downtime**
- Good for **DR-to-Cloud scenarios**

2.4 Transactional Replication

- Configure on-prem SQL Server as publisher
- EC2 SQL Server as subscriber

Praveen Madupu +91 98661 30093
Sr SQL Server DBA
praveensqldba12@gmail.com

- Cutover with minimal downtime

2.5 Always On Availability Groups (AG)

- Extend Availability Group to EC2 SQL Server
- Requires **Active Directory domain** and **Windows Failover Clustering**
- Perform manual failover when ready

2.6 SQL Server Mirroring

- Set up mirroring between on-prem and EC2
- Limited by newer SQL versions (deprecated in some)

✅ 3. Migrate to Amazon Aurora (MySQL/PostgreSQL compatible)

Not a SQL Server target, but for **heterogeneous migration**, you can convert SQL Server to Aurora.

🔹 Migration Methods:

3.1 AWS SCT + AWS DMS

- Use **Schema Conversion Tool** to convert SQL Server schema to Aurora-compatible
- Migrate data using **AWS DMS**
- Used in **modernization projects**

✅ 4. Hybrid or Staged Migration Approaches

🔹 Migration Methods:

4.1 AWS DataSync or Transfer Family

- Move large data files (backups, logs) to AWS quickly
- Support **NFS, SMB, SFTP**, etc.

Praveen Madupu +91 98661 30093
Sr SQL Server DBA
praveensqldba12@gmail.com

4.2 Third-Party Tools

- **Quest SharePlex, Redgate, Attunity, NetApp Cloud Sync**
- GUI-based, more automation, error handling, and scheduling

4.3 Custom Scripts

- Use PowerShell or T-SQL to script backup, compression, transfer, and restore

📊 Summary Table: On-Prem SQL Server to AWS Migration Options

Target AWS Service	Migration Method	Downtime	Best For
Amazon RDS for SQL Server	AWS DMS (online/offline)	Low/High	Production workloads
	Native Backup/Restore via S3	Medium	Medium-sized databases
	BACPAC Import	High	Small, non-critical DBs
	AWS Snowball	Medium	Large DBs, slow internet
EC2 (SQL Server)	Backup & Restore	Medium	Most use cases
	Log Shipping, AG, Replication	Low	Critical systems, DR scenarios
	Detach/Attach	High	Simple use cases, small DBs
Amazon Aurora	AWS SCT + DMS	Medium-High	Modernization (change engine)

📌 Notes:

- **EC2** gives full control (ideal for lift-and-shift)
- **RDS** is easier to manage (patches, backups, HA handled by AWS)
- Choose **DMS** for minimal downtime migrations
- Always assess compatibility (features, stored procs, etc.)

Praveen Madupu +91 98661 30093
Sr SQL Server DBA
praveensqldba12@gmail.com