Azure Data Studio (ADS) — what it is, what it offers, and some important context & caveats. I break it down into **What it is**, **What you get**, and **What you should be aware of**.

Reference:
https://learn.microsoft.com/en-gb/azure-data-studio/download-azure-data-studio?tabs=win-install%2Cwin-user-install%2Credhat-install%2Cwindows-uninstall%2Credhat-uninstall

✅ **What is Azure Data Studio**

- Azure Data Studio is a **lightweight, cross-platform data-management and development tool** built by Microsoft. (Microsoft Learn)
- It supports Windows, macOS, and Linux. (Microsoft Learn)
- It works with a wide variety of databases, including traditional deployments such as SQL Server (on-premises or in-cloud), Azure SQL Database, and many others (MySQL, PostgreSQL, etc.) through extensions. (Microsoft Learn)
- The tool is free to use — for personal, commercial, or enterprise use. (Microsoft Learn)

In short: ADS is meant to be a modern, cross-platform "Swiss-army knife" for querying, managing, and developing across different database systems.

✨ **What You Get — Key Features & Strengths**

Here are the main capabilities of ADS that make it attractive for developers, DBAs, and data professionals:

- **Modern SQL query editor** — Includes IntelliSense, code snippets, syntax highlighting, autocomplete, multiple tabs, and comfortable scripting capabilities. (Microsoft Learn)
- **Integrated terminal** — You can use familiar command-line tools like sqlcmd, PowerShell, Bash (depending on OS), or other CLIs directly within ADS. (Microsoft Learn)
- **Object explorer / server-database connection management** — Manage server connections, group servers, explore database schema, run queries, browse objects. (Microsoft Learn)
- **Result visualization & export** — After running queries you can view results in grid/tabular form, export to JSON, CSV, Excel, etc. (Microsoft Learn)
- **Notebooks support (Jupyter-style)** — For data analysis, experimentation, documentation: support for SQL notebooks, plus extension support for languages like Python, Spark, etc. (Microsoft Learn)
- **Extensibility via Extensions** — You can install extensions for additional databases (MySQL, PostgreSQL, etc.), migrations, charting, schema compare, Git integration, etc. (Microsoft Learn)
- **Cross-platform & open source** — Because it supports Windows, macOS, Linux, it's useful in heterogeneous environments (cloud, on-prem, hybrid). ADS is open-source (source code on GitHub) under permissive licensing. (Microsoft Learn)

⚠️ **What You Should Be Aware Of (Limitations & Lifecycle)**

There are some caveats/changes you should know about before relying on ADS long-term:

- **Retirement Notice**: ADS is scheduled to be **retired on February 28, 2026**. (Microsoft Learn)
- Because of that, Microsoft recommends using instead the MSSQL extension for Visual Studio Code (in Visual Studio Code) — which offers many of the same capabilities and will be the future-facing tool. (Microsoft Azure)
- ADS is more geared toward **querying, development, data-engineering, scripting, and lightweight admin tasks** — rather than deep server-level administration. For heavy admin, complex configuration, performance tuning, or fully-featured GUI-based administration, SQL Server Management Studio (SSMS) remains the more fully-featured tool. (Microsoft Learn)
- Some administrative features in ADS are only available via extensions, or are in "preview / limited" form (compared to SSMS). (Microsoft Learn)
- Because ADS is now slated for retirement, investing heavily in ADS-specific workflows might require retooling again in future. Better to adopt the MSSQL extension for VS Code (or stick with SSMS) for long-term systems.

🎯 **Who Should Use (or Continue Using) Azure Data Studio — And Who Should Not**

Good fit for ADS:

- Database developers and data engineers who write T-SQL, run ad-hoc queries, build scripts, or data analytics tasks.
- Teams working across OS (Windows, macOS, Linux) — e.g. developers on macOS connecting to on-prem or cloud SQL.
- Users who like a lightweight, flexible, and extensible environment with modern editor features (snippets, IntelliSense, notebook support, Git integration).
- Quick data exploration, data transformation, analysis, reporting, or working with multiple database types (SQL Server, MySQL, PostgreSQL, etc.)

When you should *not* rely solely on ADS:

- Heavy-duty database administration: server configuration, security & permissions, performance tuning, advanced maintenance tasks. SSMS remains better for that.
- When you need guaranteed long-term tool support (since ADS is being retired). For long-term maintenance or enterprise operations, plan to use VS Code + MSSQL extension or SSMS.
- Complex GUI-driven tasks such as database diagrams, advanced GUI designers, live query statistics, deep diagnostic dashboards — ADS may not offer full functionality compared to SSMS.

📦 **Summary:**

- Azure Data Studio is a **free, cross-platform, lightweight database tool** for querying, scripting, and data-development across multiple database types. (Microsoft Learn)
- It provides a **modern editing environment** (IntelliSense, snippets, multi-tab, terminal, notebooks, Git) and good support for hybrid on-prem/cloud database scenarios. (Microsoft Learn)
- However — important — **it is scheduled for retirement (Feb 28, 2026)**. (Microsoft Learn)
- For long-term or enterprise projects, Microsoft recommends switching to **VS Code + MSSQL extension** (or using SSMS for heavy administration). (Microsoft Azure)

Decision-ready comparison of the three tools:

Azure Data Studio (ADS) vs **VS Code + MSSQL Extension** vs **SQL Server Management Studio (SSMS)**.

⭐ Summary Comparison Table

Feature / Need	Azure Data Studio (ADS)	VS Code + MSSQL Extension	SQL Server Management Studio (SSMS)
Best For	Querying, scripting, notebooks, multi-OS work	Long-term SQL development on modern, supported platform	Deep SQL Server administration
OS Support	Windows, macOS, Linux	Windows, macOS, Linux	Windows only
Retirement Status	❗ Retiring Feb 28, 2026	Supported long-term	Fully supported
Query Editing	Excellent	Excellent	Good, older editor
IntelliSense	Yes	Yes	Yes
Jupyter Notebooks	✓ Built-in	Via extension	❌ No
Extensions	Many DB + data extensions	Very large extension ecosystem	Limited
Administration Tools	Basic	Basic	✓ Full (SQL Agent, replication, mirroring, PBM, XE GUI, security, etc.)
Performance Tuning	Limited	Limited	✓ Best (DMVs, XE, plans, live stats)
Migration Tools	Azure SQL Migration extension	MSSQL & SQL tools extensions	Limited
Use for SQL Server Upgrades	Yes (via extensions)	Yes	✓ Fully
Future Investment	❌ No (end-of-life)	✓ Strong	✓ Strong
Ease of Use	Very friendly	Friendly	Full-featured but heavier
Target Audience	Developers, data engineers	Developers, long-term SQL development	DBAs, SQL admins

🟦 Which One Should YOU Use? (Recommendation)

✓ If you want the future-proof and recommended platform

➞ VS Code + MSSQL Extension

- Supported long-term
- Modern UI
- Cross-platform
- Powerful extensions
- Will replace ADS concepts + features

✓ If your work is mostly administration, tuning, backups, jobs, monitoring

➞ SQL Server Management Studio (SSMS)

- Required for many admin tasks
- Best for DBAs
- Only tool that supports *complete* SQL Server GUI administration

✓ If you want SQL notebooks, quick querying, cross-platform work *right now*

➞ Azure Data Studio (ADS)

- Still great today
- But keep in mind retirement (Feb 2026)
- Gradually transition to VS Code + MSSQL extension

🟨 Detailed Feature Breakdown

1️⃣ Query Editing Experience

ADS	VS Code + MSSQL	SSMS
Modern editor	Modern editor	Good but older editor
Snippets	Snippets	Snippets
Multi-tab	Multi-tab	Multi-tab
Themes	1000+ themes	Limited themes

Winner: **ADS / VS Code**

2️⃣ Administrators (DBA) Capabilities

Task	ADS	VS Code	SSMS
SQL Agent	❌	❌	✓
AlwaysOn (AG)	❌	❌	✓
Replication	❌	❌	✓
Server configuration	❌	❌	✓
XE GUI	❌	❌	✓
Security & logins	Limited	Limited	✓
Query Store GUI	Partial	Partial	✓ Best
Graphical plan viewer	✓	✓	✓ (best)

Winner: **SSMS**

3️⃣ Migration & Assessment

Tool	Migration Support
ADS	Azure SQL Migration extension (supports SQL 2022)
VS Code	Azure SQL tools support (future-focused)
SSMS	No modern migration assessment tools built-in

Winner: **ADS / VS Code**

4️⃣ Future & Longevity

Tool	Status
ADS	❗ Retires Feb 2026
VS Code + MSSQL	✓ Long-term future
SSMS	✓ Supported indefinitely

Winner: **VS Code + MSSQL**

🟩 Final Recommendations (Straightforward)

👉 Use SSMS

- Administration
- Performance tuning
- Daily DBA tasks

👉 **Use VS Code + MSSQL extension**

- Long-term SQL development
- Query editing
- SQL modernization
- Replacement for ADS

👉 **Use ADS (short-term)**

- Notebooks
- Lightweight SQL dev
- Migration tasks (Azure SQL Migration extension)

1 DBA (Database Administrator)

Tasks like:

- Performance tuning
- Backup & restore
- Server maintenance
- SQL Agent jobs
- Indexing & troubleshooting
- AlwaysOn / high availability

2 Database Developer / SQL Developer

Tasks like:

- Writing T-SQL
- Stored procedures, functions, triggers
- Query optimization
- Schema changes
- Working with dev/test databases

3 Data Engineer / Analyst

Tasks like:

- SQL + Python + notebooks
- ETL/ELT workflows
- Data exploration
- Running analytical queries

4 Hybrid (DBA + Developer)

A mix of both:

- Administration + query development
- Maintain servers + write SQL code

Roles (DBA + Developer + Data Engineer + Hybrid), you need a tooling setup that supports:

- Full SQL Server administration
- Query development & optimization
- Data engineering workflows (notebooks, scripting)
- Migration + modernization
- Long-term compatibility
- Cross-platform support (if needed)
- and future-proofing (especially because Azure Data Studio is retiring)

Below is the **best complete toolset** with detailed guidance on what to use for each type of work.

🟩 **Your Ideal SQL Tooling Setup (All Roles Combined)**

⭐ **1. SQL Server Management Studio (SSMS)**

✓ **Best for: DBA + Performance + Administration**

SSMS is **mandatory** for anyone doing DBA tasks.

You will use SSMS for:

- SQL Agent job management
- Backups, restores, maintenance plans
- AlwaysOn / Availability Groups
- Activity Monitor, Extended Events management
- Security, auditing, server configuration
- Query Store GUI (full features)
- Execution plan deep analysis
- Replication, mirroring, etc.

🔧 **Recommendation:**

Install the **latest SSMS** and keep it updated.

⭐ **2. Visual Studio Code (VS Code) + MSSQL Extension**

✓ **Best for: Developer + Future-Proof SQL Editing**

Since Azure Data Studio will retire in February 2026, VS Code with the MSSQL extension becomes the long-term platform.

You will use VS Code for:

- T-SQL editing with IntelliSense
- Lightweight querying
- Git-based SQL source control
- SQL notebooks (using extensions)
- Multi-database work
- CI/CD integration
- Data engineering workflows

🔧 **Recommended Extensions:**

- **MSSQL** (main SQL extension)
- SQL Notebook extensions
- GitLens
- YAML / JSON tools for DevOps
- Docker (if doing containerized SQL)

⭐ 3. Azure Data Studio (ADS) — Use for Now, But Prepare to Transition

✓ Best for NOW: Notebooks, Migration, Lightweight SQL Work

Azure Data Studio is still great for current tasks:

- SQL notebooks (T-SQL + Python + Markdown)
- Azure SQL Migration extension (supports SQL Server 2022 assessments)
- Server dashboards
- Easy data exports
- Cross-platform SQL dev

BUT: 🔴 **It retires Feb 2026** → start shifting work into VS Code gradually.

⭐ 4. PowerShell + SQLServer Module

✓ Best for: Automation + DBA Scripts

Use PowerShell for:

- Automated backups/restores
- Index maintenance
- SQL Agent job automation
- Migration scripting
- Server configuration scripting
- CI/CD pipelines

Install: Install-Module SqlServer

⭐ 5. Azure CLI + Azure PowerShell (If using Azure SQL)

✓ Best for: Cloud DB operations

You can automate:

- Azure SQL deployments
- Managed Instance tasks
- Security, networking, firewall rules
- Migration pipelines

🟦 When to Use Which Tool (Clear Guide)

Task	Recommended Tool
Query tuning (simple)	VS Code or ADS
Query tuning (advanced)	SSMS
SQL Agent jobs	SSMS
Backups/restores	SSMS
AlwaysOn AG admin	SSMS
Writing stored procedures	VS Code or SSMS
Notebook analysis	ADS or VS Code
Performance troubleshooting	SSMS
Schema compare	VS Code (extension) or ADS
Migration to Azure SQL	ADS Migration Extension
CI/CD pipelines	VS Code + PowerShell
Data analysis	ADS notebooks
Cross-platform SQL work	VS Code or ADS

🟧 **Why You Need ALL Tools (and how they fit together)**

✓ **SSMS — Your DBA Tool**

Critical for deep administrative operations that no other tool replaces.

✓ **VS Code — Your Long-Term Development Platform**

Modern editing + cross-platform + future-proof solution for SQL development & data engineering.

✓ **ADS — Your Current Migration & Notebook Tool**

Great for SQL notebooks and migration work (especially SQL Server 2022 assessments), until retirement.

✓ **PowerShell — Your Automation Engine**

For serious DBA automation and DevOps integration.

🟩 **Final Recommendation (Your Complete Setup)**

Install all of the following:

1. **SQL Server Management Studio (SSMS) — required for DBA work**
2. **Visual Studio Code + MSSQL extension — long-term dev + hybrid work**
3. **Azure Data Studio (temporary) — for notebooks & migration assessments**
4. **PowerShell SQLServer module — for DBA automation**
5. **Optional:** Azure CLI + Azure PowerShell if you manage Azure SQL

✅ 1. Ideal Daily Workflow

This workflow integrates SSMS, ADS, VS Code, PowerShell, and Azure tools.

🔷 Start Your Day (Monitoring & Administration)

Use **SSMS** for:

- Checking SQL Agent jobs
- Reviewing alerts, error logs
- Monitoring blocking, waits, AG health
- Looking at performance dashboards (Query Store, Activity Monitor)

Tools:

- **SSMS** (primary)
- PowerShell (automation scripts if needed)

🔷 Development & Query Writing

Use **VS Code + MSSQL extension** for:

- Writing SQL stored procedures/functions
- Editing scripts with IntelliSense
- Using Git for version control
- Debugging/rewriting long queries
- Working across databases (SQL Server, Postgres, MySQL)

Tools:

- **VS Code**
- **MSSQL extension**
- Git / GitLens

🔷 Data Engineering / Analytics Tasks

Use **Azure Data Studio** (ADS) for:

- SQL + Python notebooks
- Ad-hoc data analysis
- Quick formatting / data visualization
- Lightweight querying
- Export to CSV/Excel
- Sharing reproducible scripts with notebooks

When ADS retires → move this into VS Code notebooks.

Tools:

- **Azure Data Studio** (today)
- **VS Code Notebook extensions** (future ready)

🔷 Migration / Modernization Work

Use **Azure Data Studio** with the **Azure SQL Migration extension** for:

- Migration readiness assessment
- SQL Server 2022 → Azure SQL migration
- Schema comparison
- Detecting compatibility issues

Tools:

- **Azure Data Studio**
- Azure SQL Migration extension
- VS Code (long-term replacement)

🔷 **DBA Automation & DevOps Pipelines**

Use **PowerShell** and/or Azure CLI for:

- Automated backups/restores
- Automated index maintenance jobs
- CI/CD deployment scripts
- Azure SQL provisioning (if cloud)

Tools:

- **PowerShell SQLServer module**
- Azure CLI
- Azure PowerShell

✅ **2. Installation Package (Everything You Need)**

Here is the exact tool list you should install:

A. SSMS (must-have for DBA)

Download latest SSMS (required for admin tasks).

- Features: Agent, AGs, XE GUI, deep tuning

B. Visual Studio Code + SQL Extensions

Install:

- VS Code
- MSSQL extension
- SQL Notebook extension
- GitLens
- XML/JSON tools
- Docker extension (optional)

C. Azure Data Studio (temporary but useful)

Install ADS for:

- SQL notebooks
- Migration extension
- Lightweight SQL dev

D. PowerShell Tools

Run:

Install-Module SqlServer

Install-Module Az

E. Azure CLI

Install Azure CLI for cloud operations.

✅ **3. Best Tool For Each Task (Quick Map)**

Task	**Best Tool**
SQL Server Administration	**SSMS**
Query Development	**VS Code**
Lightweight Querying	ADS / VS Code
Migration to Azure	ADS Migration Extension
SQL Server Upgrade Checks	ADS Migration Extension
Query Optimization (deep)	**SSMS**
SQL Agent Jobs	**SSMS**
Backups / Restores	**SSMS**
Scripting Automation	PowerShell

Task	Best Tool
Cloud SQL Management	Azure CLI + Azure PowerShell
Data Engineering	ADS Notebooks / VS Code notebooks
Cross-Platform SQL Work	**VS Code**

✅ 4. Learning / Adoption Roadmap

Phase 1: Core Tools Setup

- Install SSMS + VS Code + ADS
- Install PowerShell modules
- Configure Git + repositories
- Install migration extensions

Phase 2: Hybrid SQL Work

- Use SSMS for DBA
- Use VS Code for writing queries
- Use ADS for notebooks & migration

Phase 3: Transition to Long-Term Tools

- Migrate ADS notebooks → VS Code
- Migrate SQL connections to VS Code
- Reduce reliance on ADS (retiring 2026)

Phase 4: Automate Your Work

- Build PowerShell scripts
- Automate backups, jobs, monitoring
- Use Azure CLI for cloud integration

SQL tooling package, including downloads, installation steps, extensions, workflows, scripts, templates, and checklists.
This is everything a **DBA + Developer + Data Engineer + Hybrid** professional needs.

🟩 SECTION 1 — DOWNLOAD LINKS (ALL TOOLS)

✅ 1. SQL Server Management Studio (SSMS)

SSMS 20.2.1 version: https://aka.ms/ssmsfullsetup

- Required for SQL Agent, AGs, deep tuning, admin tasks.

✅ 2. Visual Studio Code (VS Code)

Download: https://code.visualstudio.com/download

- Long-term replacement for ADS
- Best editor for SQL development

VS Code Key Extensions

After install, press **Ctrl+Shift+X → Search & install:**

SQL extensions

- **MS SQL** (Microsoft SQL Server extension)
- SQL Notebook (via Python or .NET Jupyter support)

Productivity

- GitLens
- YAML
- JSON Tools
- Docker (optional)
- PowerShell extension

✅ 3. Azure Data Studio (ADS)

Download: https://aka.ms/azuredatastudio
(Still useful until retirement in Feb 2026)

ADS Extensions to install:

- **Azure SQL Migration**
- Schema Compare
- SQL Database Projects
- Query Performance Insight
- SQLite/PostgreSQL/MySQL (optional)

✅ 4. PowerShell Modules

Run in **PowerShell as admin**:

```
Install-Module SqlServer -Scope AllUsers
Install-Module Az -Scope AllUsers
Install-Module dbatools -Scope AllUsers
```

✅ 5. Azure CLI

Download: https://learn.microsoft.com/cli/azure/install-azure-cli

🟩 SECTION 2 — INSTALLATION GUIDE (STEP BY STEP)

⭐ Step 1 — Install SSMS

1. Run SSMS installer
2. Install latest SQL Server tools

3. Restart computer
4. Open SSMS → Connect to server

⭐ **Step 2 — Install VS Code + Extensions**

1. Install VS Code
2. Open it → Press **Ctrl+Shift+X**
3. Install extensions:
    - MSSQL
    - GitLens
    - PowerShell
    - Python (optional, for notebooks)
4. Restart VS Code

⭐ **Step 3 — Install Azure Data Studio**

1. Install ADS
2. Open → Extensions → Install:
    - Azure SQL Migration
    - Schema Compare
    - SQL Projects

⭐ **Step 4 — Install PowerShell modules**

In **PowerShell**:

```
Install-Module SqlServer
Install-Module Az
Install-Module dbatools
```


⭐ **Step 5 — Configure Git (Optional but recommended)**

```
git config --global user.name "Your Name"
git config --global user.email "Your Email"
```

🟩 **SECTION 3 — DAILY WORKFLOWS (ALL ROLES)**

🔷 **1. DBA Daily Workflow**

Use **SSMS**:

- Check SQL Agent job failures
- Check AlwaysOn AG dashboard
- Review error logs
- Check blockings
- Check CPU/memory waits
- Review backups completed

PowerShell automation:

```
Get-SqlAgentJob -ServerInstance "MyServer" | Where LastRunOutcome -ne 'Succeeded'
```

🔷 **2. Developer Daily Workflow**

Use **VS Code**:

- Write stored procedures
- Test queries
- Use Git for versioning
- Work with multiple DBs
- Deploy dev/test databases

🔷 3. Data Engineer / Analyst Workflow

Use **Azure Data Studio**:

- SQL notebooks (SQL + Python)
- CSV/Excel export
- Data cleaning
- Export data to files
- Quick visualizations

🔷 4. Migration / Upgrade Workflow

Use **Azure Data Studio + Migration Extension**:

- Run SQL Server 2022 readiness checks
- Identify incompatible features
- Generate assessment report
- Migrate to Azure SQL or SQL MI

🟩 SECTION 4 — SCRIPT PACKS (READY TO RUN)

⭐ SQL Server Health Check

```
SELECT
   wait_type, wait_time_ms, signal_wait_time_ms, waiting_tasks_count
FROM sys.dm_os_wait_stats
ORDER BY wait_time_ms DESC;
```

⭐ Check database properties

```
SELECT
   name, compatibility_level, recovery_model_desc, state_desc
FROM sys.databases;
```


⭐ Rebuild fragmented indexes

```
DECLARE @RebuildPct INT = 30;
SELECT
   'ALTER INDEX [' + i.name + '] ON [' + s.name + '].[' + t.name + '] REBUILD;'
FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'SAMPLED') AS ips
JOIN sys.indexes i ON ips.index_id = i.index_id AND ips.object_id = i.object_id
JOIN sys.tables t ON i.object_id = t.object_id
JOIN sys.schemas s ON t.schema_id = s.schema_id
WHERE avg_fragmentation_in_percent > @RebuildPct;
```

⭐ PowerShell Backup Script (dbatools)

```
Get-DbaDatabase -SqlInstance "MyServer" |
Backup-DbaDatabase -Path "D:\Backups"
```

🟩 SECTION 5 — NOTEBOOK TEMPLATE (for ADS or VS Code)

Notebook Sections:

1. **Environment Info**
2. **Server Health Checks**
3. **Database Sizes**
4. **Index Fragmentation**
5. **Top CPU Queries**
6. **Blocking Queries**
7. **Storage Space**
8. **TempDB Usage**

SECTION 6 — CHECKLISTS (DBA, Developer, Migration)

DBA Daily Checklist

- Check SQL Agent jobs
- Check AlwaysOn AG health
- Review error logs
- Monitor blocking
- Check free disk space
- Validate backups

Developer Daily Checklist

- Pull latest Git changes
- Run unit tests on procedures
- Format SQL code
- Review execution plans
- Deploy dev/test scripts

Migration Checklist

- Run Azure Data Studio Migration Assessment
- Fix deprecated features
- Validate compatibility level
- Validate linked servers
- Export assessment report
- Perform test migration
- Perform production migration

SECTION 7 — PROJECT FOLDER STRUCTURE

Use this recommended structure:

```
SQL-Projects/
|
├── Admin/
|   ├── maintenance/
|   ├── backups/
|   └── health-checks/
|
├── Development/
|   ├── StoredProcedures/
|   ├── Functions/
|   ├── Tables/
|   └── Views/
|
├── Migration/
|   ├── assessments/
|   ├── scripts/
|   └── reports/
|
└── Notebooks/
    ├── performance/
    ├── admin/
    └── analysis/
```

🟩 SECTION 8 — FUTURE-PROOF STRATEGY

Since ADS is retiring, here is the roadmap:

2024–2025

- Use ADS for notebooks + migration
- Use VS Code for SQL development
- Use SSMS for DBA tasks

2026 onward

- Move all ADS tasks → VS Code notebooks
- Use MS SQL extension
- Continue using SSMS for administration

🟩 All-In-One Summary

You now have:

✓ Tools
✓ Installation scripts
✓ Extensions
✓ DBA workflow
✓ Developer workflow
✓ Data engineer workflow
✓ Migration workflow
✓ SQL scripts
✓ PowerShell automation
✓ Project structure
✓ Checklists
✓ Future-proof plan

Full Onboarding Guide for Your Team

Purpose: Ensure new DBAs, developers, and analysts get up to speed quickly.

A. Tool Installation

1. Install **SSMS** (admin tasks)
2. Install **VS Code + MSSQL extension** (development)
3. Install **Azure Data Studio** (notebooks, migration)
4. Install **PowerShell modules** (SqlServer, Az, dbatools)
5. Install **Azure CLI** (cloud tasks)

B. Environment Setup

- Configure Git (git config --global user.name/email)
- Create folder structure:

```
SQL-Projects/
├── Admin/
├── Development/
├── Migration/
└── Notebooks/
```

- Map network drives for backups and reports

C. Daily Workflow

1. **DBA tasks:** Check SQL Agent jobs, backups, performance dashboards (SSMS)
2. **Developer tasks:** Write, test, and commit T-SQL scripts (VS Code)
3. **Data engineering tasks:** Run notebooks for analysis (ADS or VS Code)
4. **Migration:** Assess databases with ADS Migration extension

D. Automation

- Use SQL_AutomationToolkit.ps1 for backups, health checks, indexing
- Schedule as Windows Task or SQL Agent Job

E. Documentation & Versioning

- Keep SQL scripts in Git repository
- Document major changes in CHANGELOG.md
- Use notebook templates for repeatable analysis

🟩 4️⃣ Complete Migration Runbook

Purpose: Step-by-step guide to migrate SQL Server (on-prem) → Azure SQL or SQL Managed Instance.

A. Pre-Migration

1. Inventory all databases
2. Run **Azure Data Studio Migration Assessment**
3. Check **compatibility levels**
4. Identify deprecated features
5. Validate security / logins / users

B. Test Migration

1. Restore database to test environment
2. Apply schema changes / fixes
3. Test queries, jobs, stored procedures
4. Compare source vs target data

C. Production Migration

1. Backup production database (full backup)
2. Stop non-essential services or schedule downtime
3. Restore database to target Azure SQL / SQL MI
4. Apply logins, permissions, jobs
5. Run validation scripts to confirm success

D. Post-Migration

1. Monitor CPU, IO, TempDB usage
2. Validate scheduled jobs
3. Update connection strings in applications
4. Document migration in change log

E. Rollback Plan

- Keep full backup from pre-migration
- If migration fails, restore backup to source
- Notify stakeholders and perform root cause analysis

Word Doc: https://github.com/PMSQLDBA/SSMS-Studio/blob/main/Azure%20Data%20Studio%20--%20Information.docx