

Visual Studio for SQL Developers

September 1, 2015 Copenhagen, Denmark

Andrey Zavadskiy, Krasnodar, Russia MCSE/MCSD/MCT

About me

- Solutions architect, SQL & .NET developer
- 20 years in IT industry
- Worked with SQL Server since7.0 back in 2001
- Developed in C#, ASP.NET, MVC, JavaScript, SharePoint
- MCDBA, MCSE, MCSD
- MCT since 2008
- PASS speaker



- http://andreyzavadskiy.com
- https://www.facebook.com/ andrey.k.zavadskiy
- @AndreyZavadskiy
- https://www.linkedin.com/in/ zavadskiy



About Krasnodar

Regional center
Was founded in 1793, renamed in 1920
Original name Yekaterinodar – Catherine's gift

Distances:

Istanbul 929 km

Moscow 1196 km

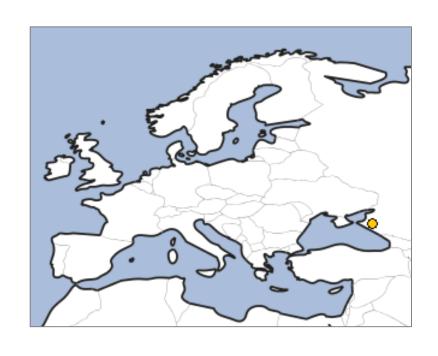
Warsaw 1541 km

Copenhagen 2200 km

Brussels 2640 km

Paris 2793 km

Lisbon 3995 km





Session Goal

Provide a practical overview of how to use SQL Server Data Tools to create, test, deploy and debug a database project



Contents

- Online database development
- Offline database development
- Deployment
- Debugging
- Database unit testing
- Additional tools



Concept of SQL Server Data Tools

- Focusing on the final version of code rather then on many ALTER iterations
- The unique IDE for all database developer's needs
 - Based on Visual Studio
 - Working with connected databases (like SQL Server Management Studio)
 - Project approach to code writing
 - Debugging and unit testing
 - Version control (via integration with Team Foundation Server)



Online Database Development

- Available through SQL Server Object Explorer
- Functions are similar to SQL Server Management Studio
- Tools:
 - Table Designer
 - Code Editor
 - Visual Data Editor
 - Query Window
 - Scripting capabilities



Working with connected database



Offline Database Development

- Creating a database project
- Creating database objects
 - Import from existing database
 - Creating from scratch
- Project properties



Creating a database project



Database Deployment

- Deployment in the connected database
 - Publishes the script (CREATE)
 - Compares against the target database
 - Creates a change script (ALTER)
 - Runs the change script on the target database
- Deployment in the disconnected database
 - Creates a DACPAC package
 - Distribute and publish DACPAC
- Publishing profiles and settings



Publishing a database



Deployment Scripts For Inserting Data

Creating scripts

- Based on the existing data
- From scratch

Various types of scripts

- T-SQL script
- Pre-deployment
- Post-deployment



Scripting data for deployment



Multiple Databases

Using objects from another database on:

- Same instance
- Another instance needs linked server
- Addressing remote database object:
 - 4-part naming convention
 - Synonyms
- SSDT knows nothing about remote database objects

Deployment to multiple databases

 Impossible, can be implemented with multiple projects and multiple deployments



Using data from other databases



Debugging

- Only in connected environment
- Breakpoint can be set with SQL Server Object Explorer

Two modes:

- 1. Execute query with debugger
 - Similar to SQL Server Management Studio
- 2. Classic Visual Studio debugging with F5
 - Has a separate connection to SQL Server/database
 - Needs a project startup script



Debugging T-SQL code



Database Unit Testing

Unit test

- Is performed on a smallest piece of testable code
- Isolated from the other pieces of code
- Should be repeatable
- Gives the answer to only one question
- Usually created by developers



What for?

- Confidence in your code
- Confirms that product requirements are working
- Early error checking of code
- Instant visual feedback on errors
- Helps to check subsequent changes in code
- Provides documentation for other developers



Where are the bugs?

Dependency





What can be tested?

Meta-data

- Table structure, field type and length
- Existence of objects

Constraints

CHECK, DEFAULT, PRIMARY KEY, FOREIGN KEY, UNIQUE

T-SQL code

Stored procedures, Functions, Triggers

Security permissions

Execution time



Data that can be tested

Scalar values

- Normal values
- Errors (incorrect values)
- Very big values
- NULL

Table values

- Rowset
- Empty rowset
- Very big rowset
- Metadata



- Creating a database unit test project
- Unit Test Project Internals



Unit Test Flow

Test initialize Unit test(s)

- Pre-test
- Test
- Post-test

Test cleanup



Unit Test Features

- Can have more than one test condition
- Can handle exceptions raised in database
- Can be run within a transaction
- Can use a second connection for pre/post test phases
- Allows to create and use custom test conditions as Visual Studio extension (dll)
 - How-to: https://msdn.microsoft.com/en-us/library/jj860449(v=vs.103).aspx
 - Example: https://ssdtconditions.codeplex.com/



- Positive test
- Negative test
- Running test in transaction
- Checking metadata
- Checking table equality



Debugging in unit tests

- Can debug only the T-SQL code to be tested
 - Breakpoint can be set inside the stored procedure, function or trigger
- Can't debug the T-SQL code of the unit test itself



Debugging a code from within a database unit test



Additional tools

- Comparison tools
 - Only in Professional and Ultimate Editions
 - Schema compare
 - Data compare
 - Needs a key
- Code analysis
- Refactoring



Comparing schemas and data



Summary

- SSDT unique tool for SQL developer
- Declarative model of database development
- Covers all stages:
 - Create
 - Debug
 - Test
 - Deploy



References

MSDN: SQL Server Data Tools

https://msdn.microsoft.com/enus/library/hh272686(v=vs.103).aspx

SSDT Team Blog

http://blogs.msdn.com/b/ssdt/

MSDN Forum

https://social.msdn.microsoft.com/Forums/sqlserver/en-US/home?forum=ssdt





Questions?





Thank you for attending!