T-SQL Programming
This course covers Microsoft’s SQL Server database programming techniques using Transact-SQL programming constructs. Using coding structures like stored procedures, functions, and triggers, it is possible to achieve high levels of capability, flexibility, and security in your Microsoft SQL Server database. The course will provide students with hands-on practice writing scripts and procedures to reinforce these programming concepts.

Who should take this course?
This course is designed for students who want to develop skills using Microsoft’s Transact-SQL language in an administrator, developer, or analyst role. It is expected that the student is already familiar with the SQL language and the course will build on that knowledge. This course is a prerequisite for taking more advanced courses such as SQL Server Reporting Services (SSRS).

Course Objectives
• Automate data insertion and modification using scripts and basic programming constructs.
• Convert SQL Scripts into procedures and functions to make the functionality available to other database users.
• Manage transactions in order to increase concurrent usage of the database.
• Create SQL programming functions to provide flexible data integrity and error-handling rules.
• Create triggers that enable the SQL database to automatically take action based on a particular event occurring.
• Manage database query results row-by-row using SQL cursors.

Course Details
• Length: 33 hours
• Format: Classroom
• Prerequisites: Structured Query Language: (T-SQL) Level 2 or equivalent

The above prerequisites are considered to be the basic skills and knowledge needed prior to taking this class. Instructors will assume your readiness for the class materials and will NOT use class time to discuss prerequisite materials.
Course Contents

Automate data insertion and modification using scripts and basic programming constructs.

• Create scripts to perform insert and update tasks within the database.
• Add flow-control statements and logic to a script to perform branched processing based on conditions.
• Code loops that make the script execute the workflow repeatedly over a series of data.
• Build in error handling routines using Try…Catch clauses.

Convert SQL Scripts into procedures and functions to make the functionality available to other database users.

• Transform scripts into packaged procedures and functions that can be stored in the database and executed as required.
• Use input and output parameters to pass information “on-the-fly” to and from the procedure or function.
• Use the built-in debugging tool to help find errors in newly written procedures and functions.
• Integrate assemblies from other programming languages
• Activate regular expression support using SQL CLR functionality.

Manage transactions in order to increase concurrent usage of the database.

• Create transaction-based workflows to ensure all required work is executed completely as a unit and does not produce any errors.
• Resolve issues related to lock contention and deadlock.
• Apply best practices for T-SQL programming to use transactions effectively and minimize potential concurrency pitfalls.

Create SQL programming functions to provide flexible data integrity and error-handling rules.

• Create constraints to provide basic error checking on user entered information.
• Code T-SQL scripts to provide advanced tests for the correctness of information entered into the database by users.
Course Contents, continued

Create triggers that enable the SQL database to automatically take action based on a particular event occurring.

- Transform a script into a trigger which will execute automatically when the user makes a modification (i.e., insert, update, or delete) to the database.
- Create code that will enable and disable triggers as well as control when they execute within a given workflow.
- Build an audit trail of database inserts, updates, and deletes using triggers to keep track of changes and previous values.

Manage database query results row-by-row using SQL cursors.

- Declare cursors to interact with the data one record at a time.
- Interact with the cursor using a loop to fetch data.
- Navigate through the cursor results backward and forward in a scrollable fashion.
- Detect underlying data changes using a cursor.