

Course 2784 — Instructor-led

Course Length: 3 days

At the end of this course, students will be able to:

- Normalize databases.
- Design a normalized database.
- Optimize a database design by denormalizing.
- Optimize data storage.
- Manage concurrency
- Manage concurrency by selecting the appropriate transaction isolation level.
- Select a locking granularity level.
- Optimize and tune queries for performance.
- Optimize an indexing strategy.
- Decide when cursors are appropriate.
- Identify and resolve performance-limiting problems.

Prerequisites:

Before attending this class, students must have:

- Have working knowledge of data storage. Specifically, knowledge about row layout, fixed length field placement and varying length field placement.
- Be familiar with index structures and index utilization. Specifically, they must understand the interaction between non-clustered indexes, clustered indexes and heaps. They must know why a covering index can improve performance.
- Have had hands-on database developer experience. Specifically, three years of experience as a full-time database developer in an enterprise environment.
- Be familiar with the locking model. Specifically, students should have an understanding of lock modes, lock objects and isolation levels and be familiar with process blocking.
- Understand Transact-SQL syntax and programming logic. Specifically, students should be completely fluent in advanced queries, aggregate que-

ries, subqueries, user-defined functions, cursors, control of flow statements, CASE expressions, and all types of joins.

- Be able to design a database to third normal form (3NF) and know the trade offs when backing out of the fully normalized design (denormalization) and designing for performance and business requirements in addition to being familiar with design models, such as Star and Snowflake schemas.
- Have strong monitoring and troubleshooting skills, including using monitoring tools.
- Have basic knowledge of the operating system and platform. That is, how the operating system integrates with the database, what the platform or operating system can do, and how interaction between the operating system and the database works.
- Have basic knowledge of application architecture. That is, how applications can be designed in three layers, what applications can do, how interaction between the application and the database works, and how the interaction between the database and the platform or operating system works.
- Know how to use a data modeling tool.
- Be familiar with SQL Server 2005 features, tools, and technologies.
- Have a Microsoft Certified Technology Specialist: Microsoft SQL Server 2005 credential - or equivalent experience.

In addition, it is recommended, but not required, that students have completed:

- Course 2778, Writing Queries Using Microsoft SQL Server 2005 Transact-SQL.
- Course 2779, Implementing a Microsoft SQL Server 2005 Database.
- Course 2780, Maintaining a Microsoft SQL Server 2005 Database.

Course Outline

Unit 1: Measuring Database Performance

This unit provides students with an opportunity to measure database performance and identify database performance bottlenecks. Students will use a sample script to identify performance and concurrency problems, capture baseline performance, and prioritize identified problems for optimization.

Topics

- Importance of Benchmarking
- Key Measures for Query Performance: Sysmon
- Key Measures for Query Performance: Profiler
- Guidelines for Identifying Locking and Blocking

Lab: Measuring Database Performance

- Reviewing Tables and Scripts
- Determining Performance Baselines

Tuning and Optimizing Queries using Microsoft SQL Server 2005

Course 2784 — continued

- Prioritizing Identified Problems

After completing this unit, students will be able to:

- Describe best practices for measuring performance.
- Describe the key Sysmon counters for problem identification.
- Describe the key Profiler trace events for problem identification.
- Select best methods for identifying which procedures are causing locking and blocking.
- Review database tables and scripts.
- Use a script to identify performance and concurrency problems.
- Capture baseline performance.
- Prioritize identified problems.

Unit 2: Optimizing Physical Database Design

In this unit, students work with strategies for optimizing physical database design. Students will optimize a database schema using normalization, generalization, and denormalization.

Topics

- Performance Optimization Model
- Schema Optimization Strategy: Keys
- Schema Optimization Strategy: Responsible Denormalization
- Schema Optimization Strategy: Generalization

Lab: Optimizing Schemas

- Optimizing Memberships
- Optimizing Events
- Normalizing Event Sponsorships
- Denormalizing Membership Visits
- Cleaning Up Schema
- Adapting the Solution to the New Database Schema
- Determining Performance

After completing this unit, students will be able to:

- Explain the strategy for database optimization presented in the Performance Optimization Model.
- Explain the importance of schema design in database optimization.
- Describe the strategic use of natural and surrogate keys and their roles in schema optimization.
- Describe responsible denormalization and the role of this strategy in schema optimization.
- Describe generalization and the role of this strategy in schema optimization.

- Normalize a database schema for optimization.
- Generalize a database schema for optimization.
- Denormalize a database schema for optimization.
- Clean up database schema by verifying and adjusting data types and verifying referential integrity.
- Convert data to the new schema.
- Correct table and column names in queries, stored procedures, and triggers to reconcile schema changes.
- Test for performance.

Unit 3: Optimizing Queries for Performance

In this unit students experience optimizing and tuning queries to improve performance. In the lab, students will optimize stored procedures, views, and non-cursor aggregate queries to improve database performance and user experience.

Each query that is optimized improves the overall system because the query will use fewer resources, freeing up those resources for other queries, and reducing the amount of locking done by the query. The domino effect is profound.

Topics

- Performance Optimization Model: Queries
- What Is Query Logical Flow?
- Considerations for Using Subqueries
- Guidelines for Building Efficient Queries

Lab: Optimizing Queries

- Optimizing and Rewriting Slow Performing Stored Procedures
- Optimizing and Rewriting Slow Performing Views
- Optimizing and Rewriting Slow Performing Non-Cursor Aggregate Queries
- Determining Performance

After completing this unit, students will be able to:

- Explain the importance of set-based solutions in database optimization.
- Explain the utility of the query logical flow diagram in query optimization.
- Discuss considerations when using subqueries in query optimization.
- Describe strategies for building efficient queries.
- Rewrite stored procedures for optimization.
- Rewrite views for optimization.
- Rewrite non-cursor aggregate queries for optimization.
- Test queries for performance.

Tuning and Optimizing Queries using Microsoft SQL Server 2005

Course 2784 — continued

Unit 4: Refactoring Cursors into Queries

In this unit, students will work with strategies for refactoring cursors into queries. In the lab, students will work to optimize a database by replacing slow iterative code with faster set-based code.

Topics

- Performance Optimization Model: Query-Set-based solutions
- Five Steps to Building a Cursor
- Strategies for Refactoring Cursors

Lab: Refactoring Cursors into Queries

- Refactoring the pMembershipCategory Cursor
- Refactoring the pCommunityImpact Cursor
- Refactoring the pMemberInvitation Cursor
- Determining Performance

After completing this unit, students will be able to:

- Explain the importance of set-based solutions in database optimization.
- List five steps to building a cursor.
- Describe strategies for refactoring cursors.
- Refactor cursors into queries by rebuilding cursor logic as multiple queries, user-defined function, and complex queries with case expression.
- Test queries for performance.

Unit 5: Optimizing an Indexing Strategy

In this unit, students will work on optimizing indexing strategies. Students will work with a given database to add and delete indexes, by providing the optimum bridge between the query and the data without any redundancies.

Topics

- Performance Optimization Model: Indexes
- Considerations for Using Indexes
- Best Uses of the Clustered Index
- Best Practices for Non-Clustered Index Design
- How to Document an Indexing Strategy

Lab: Optimizing an Indexing Strategy

- Identifying Tables to Optimize
- Designing Indexes
- Determining Performance

After completing this unit, students will be able to:

- Explain the importance of optimizing index strategies in database optimization.
- Explain considerations for using indexes in data-

base optimization.

- Describe the best uses of clustered indexes as they relate to optimization.
- Describe the best practices for designing non-clustered databases.
- Explain the methodology for using an indexing strategy worksheet.
- Determine tables that need to be optimized based on slow running code.
- Design, implement, and adjust clustered and non-clustered indexes.
- Test for performance.

Unit 6: Managing Concurrency

This unit provides students with the opportunity to work with concurrency management. Students will look for concurrency issues and then solve them by optimizing transactions and adjusting the transaction isolation level.

Topics

- Performance Optimization Model: Locking and Blocking
- Multimedia - "How to Use Efficient Queries to Reduce Locking and Blocking"
- Strategies to Reduce Locking and Blocking

Lab: Reducing Blocking

- Identifying Code with Locking and Blocking Issues
- Reducing Concurrency Issues
- Determining Final Performance

After completing this unit, students will be able to:

- Explain the importance of concurrency management in database optimization.
- Explain how efficient queries reduce locking and blocking.
- List strategies for reducing locking and blocking.
- Identify code with locking and blocking issues.
- Tighten and optimize logical transactions in code with concurrency issues.
- Adjust transaction isolation levels in code with concurrency issues.
- Test database for performance.
- Determine percentage gain on database performance from baseline.