Exam 70-464: Developing Microsoft SQL Server Databases – Skills Measured

Audience Profile

This exam is intended for database professionals who build and implement databases across organizations and who ensure high levels of data availability. Their responsibilities include creating database files, data types, and tables; planning, creating, and optimizing indexes; ensuring data integrity; implementing views, stored procedures, and functions; and managing transactions and locks.

Skills Measured

NOTE: The bullets that appear below each of the skills measured are intended to illustrate how we are assessing that skill. This list is not definitive or exhaustive.

NOTE: In most cases, exams do NOT cover preview features, and some features will only be added to an exam when they are GA (General Availability).

Implement database objects (30–35%)

Create and alter tables

- develop an optimal strategy for using temporary objects, including table variables and temporary tables; define alternatives to triggers; define data version control and management; implement @Table and #table appropriately; create calculated columns; implement partitioned tables, schemas, and functions; implement column collation; implement online transaction processing (OLTP); implement columnstore and sparse columns

Design, implement, and troubleshoot security

- implement data control language statements appropriately, troubleshoot connection issues, implement execute as statements, implement certificate-based security, create loginless users, define appropriate database roles and permissions, implement contained users, implement cross db ownership chaining, implement schema security, implement server roles

Design the locking granularity level

- choose the right lock mechanism for a given task; handle deadlocks; design index locking properties; fix locking and blocking issues; analyze a deadlock scenario; design appropriate isolation level, including Microsoft ActiveX data objects defaults; design for locks and lock escalation; design transactions that minimize locking; reduce locking contention; identify bottlenecks in data design; design appropriate concurrency control, such as pessimistic or optimistic

Implement indexes

- inspect physical characteristics of indexes and perform index maintenance; identify unused indexes; implement indexes; optimize indexes, including full, filter, statistics, and force

Implement data types

- select appropriate data types, including BLOBs, GUIDs, XML, and spatial data; develop a Common Language Runtime (CLR) data type; implement appropriate use of @Table and #table; determine values based on implicit and explicit conversions

Create and modify constraints

- create constraints on tables, define constraints, modify constraints according to performance implications, implement cascading deletes, configure constraints for bulk inserts

Implement programming objects (15-20%)

Design and implement stored procedures

- create stored procedures and other programmatic objects; implement different types of stored procedure results; create a stored procedure for the data access layer; analyze and rewrite procedures and processes; program stored procedures by using T-SQL and CLR; implement parameters, including table valued, input, and output; implement error handling, including TRY…CATCH; configure appropriate connection settings

Design T-SQL table-valued and scalar functions

- modify scripts that use cursors and loops into a SET-based operation, design deterministic and non-deterministic functions

Create, use, and alter user-defined functions (UDFs)

- implement deterministic or non-deterministic functions; implement CROSS APPLY by using UDFs; implement CLR functions

Create and alter views

- set up and configure partitioned tables and partitioned views; create indexed views

Design database objects (25–30%)

Design tables

- apply data design patterns; develop appropriately normalized and de-normalized SQL tables; design transactions; design views; implement GUID as a clustered index appropriately; design temp tables appropriately, including # vs. @; design an encryption strategy; design table partitioning; design a BLOB storage strategy, including filestream and filetable; design tables for In-Memory OLTP

Design for concurrency

- develop a strategy to maximize concurrency; define a locking and concurrency strategy; design a transaction isolation strategy, including server database and session; design triggers for concurrency

Design indexes

- design indexes and data structures; design filtered indexes; design an indexing strategy, including column store, semantic indexes, and INCLUDE; design statistics; assess which indexes on a table are likely to be used, given different search arguments (SARG); design spatial and XML indexes

Design data integrity

- design a table data integrity policy, including checks, primary key, foreign key, uniqueness, XML schema, and nullability; select a primary key

Design for implicit and explicit transactions

- manage transactions; ensure data integrity by using transactions; manage distributed transaction escalations; design savepoints; design error handling for transactions, including TRY, CATCH, and THROW

Optimize and troubleshoot queries (25–30%)

Optimize and tune queries

- tune a poorly performing query, including avoiding unnecessary data type conversions; identify long-running queries; review and optimize code; analyze execution plans to optimize queries; tune queries using execution plans and Microsoft Database Tuning Advisor (DTA); optimize queries using pivots and common table expressions (CTE); design database layout to optimize queries; implement query hints; tune query workloads; implement recursive CTE; implement full text and semantic search; analyze execution plans; implement plan guides

Troubleshoot and resolve performance problems

- interpret performance monitor data; integrate performance monitor data with SQL Traces

Optimize indexes

- develop an optimal strategy for clustered indexes; analyze index usage; optimize indexes for workload, including data warehousing and OLTP; generate appropriate indexes and statistics by using INCLUDE columns; create filtered indexes; implement full-text indexing; implement columnstore indexes; optimize online index maintenance

Capture and analyze execution plans

- collect and read execution plans, create an index based on an execution plan, batch or split implicit transactions, split large queries, consolidate smaller queries, review and optimize parallel plans

Collect performance and system information

- monitor performance using Dynamic Management Views, collect output from the Database Engine Tuning Advisor, design Extended Events Sessions, review and interpret Extended Event logs; optimize Extended Event session settings, use Activity Monitor to minimize server impact and determine IO bottlenecks, monitor In-Memory OLTP resources