70-464:
Developing Microsoft SQL Server Databases

The following tables show where changes to exam 70-464 have been made to include updates that relate to SQL Server 2014 tasks. These changes are effective as of April 24, 2014.

1. Implement database objects (30-35%)

Tasks currently measured	Tasks Added/Changed post *April 2014*
Create and alter tables (complex statements) Develop an optimal strategy for using temporary objects (table variables and temporary tables); how not to rely on triggers solely as a means to manage a table; data version control and management; create tables without using the built-in tools; understand the difference between @Table and #table	**Added sub-tasks:** • create calculated columns • implement partitioned tables, schemas, and functions • implement column collation • implement in-memory OLTP
Design, implement, and troubleshoot security Grant, deny, revoke; unable to connect; execute as; certificates; loginless user; database roles and permissions; contained users; change permission chains	**Added sub-tasks:** • implement cross db ownership chaining • implement schema security • implement server roles • review effective permissions • troubleshoot and repair orphaned users
Design the locking granularity level Choose the right lock mechanism for a given task, handling and/or avoiding deadlocks; fix locking and blocking issues caused by previous development or third-party apps; analyze a deadlock scenario to alleviate the issue; impact of isolation level and ado defaults; impact of locks and lock escalation; reduce locking scenarios; how isolation levels affect blocking and locking; identify bottlenecks in, and improve, the data design	**Added sub-tasks:** • design index locking properties • design transactions that minimize locking • design appropriate concurrency control, such as pessimistic or optimistic
Maintain indexes Inspect physical characteristics of indexes and perform index maintenance; identify fragmented indexes; identify unused indexes; implement indexes; defrag/rebuild indexes; set up a	**Added sub-tasks:** • align indexes on partitioned tables • inspect indexes by using dynamic management objects

maintenance strategy for indexes and statistics; optimize indexes (full, filter index); statistics (full, filter) force or fix queue; when to rebuild versus reorg and index; create a tuning and maintenance strategy for proactive operations	
Implement data types Use appropriate data types; develop a CLR data type; understand the difference between @Table and #table; impact of GUID (newid, newsequentialid) on database performance, indexing and privacy; use spatial data; LOB data types; understand when and how to use column store and sparse columns; implicit and explicit conversions, integer math	No changes
Create and modify constraints (complex statements) Create constraints on tables; define constraints; performance implications	**Added sub-tasks:** • implement cascading deletes • configure constraints for bulk inserts
Work with XML data Implement XML; use XML (Query, Input, Output); transform XML data into relational data; retrieve relational data as XML; FOR XML; design a strategy to transform XML into relational data; design a strategy to query and modify XML data; understand XML data types and their schemas and interoperability, limitations, and restrictions; implement XML schemas and handling of XML data; how to handle it in SQL Server and when and when not to use it, including XML namespaces; import and export XML	**Added sub-tasks:** • return tables from XML data types using XQuery • implement XML selective indexes

2. Implement programming objects (20-25%)

Tasks Currently Measured	Tasks Added/Changed post *April 2014*
Write automation scripts Automate backup testing; shrink file; check index fragmentation; archive data; run an SQL Server Integration Services (SSIS) job; check disk space; automate backups	**Modified sub-task:** • Write scripts that automate backups, including backup to Windows Azure Blob Storage Service
Design and implement stored procedures Create stored procedures and other programmatic objects; techniques for developing stored procedures; different types of stored procedure results; create stored procedure for data access layer; analyze and rewrite procedures and processes; program stored procedures, with T-SQL and CLR#; use table valued parameters; encryption	**Added sub-tasks:** • implement error handling, including TRY…CATCH • configure appropriate connection settings • design appropriate query paging, including OFFSET and FETCH
Design T-SQL table-valued and scalar functions Ensure code non regression by keeping consistent signature for procedure, views and function (interfaces); turn scripts that use cursors and loops into a SET based operation	**Revised task – new full definition:** • 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) Understand deterministic, non-deterministic functions; use cross apply with UDFs; Common Language Runtime (CLR)	No Change
Create and alter views (complex statements) Set up and configure partitioned tables and partitioned views; design a best practice for using views and stored procedures and remove the direct usage of tables	**Revised task – new full definition:** • set up and configure partitioned tables and partitioned views • create indexed views

3. Design database objects (20-25%)

Tasks Currently Measured	Tasks Added/Changed post *April 2014*
Design tables Data design patterns; develop normalized and de-normalized SQL tables; understand the difference between physical tables, temp tables, temp table variables, and common table expressions; design transactions; design views; describe advantages / disadvantages of using a GUID as a clustered index; understand performance implications of # versus @ temp tables and how to decide which to use, when, and why; use of set-based rather than row-based logic; encryption (other than TDE); table partitioning; filestream and filetable	**Added sub-task:** • design tables for In-Memory OLTP
Design for concurrency Develop a strategy to minimize concurrency; handle concurrency to minimize locking and eliminate as much blocking as possible, and to avoid deadlocks; manage the transactions to limit the time to hold lock and have fast transactions (maximize concurrency); define locking and concurrency strategy; impact of read committed snapshot / snapshot isolation; understand what it solves and what it costs	**Revised task – new full definition:** • 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
Create and alter indexes Create indexes and data structures; create filtered indexes; create an indexing strategy; design and optimize indexes; design indexes and statistics; assess which indexes on a table are likely to be used given different search arguments (SARG); column store indexes; semantic indexes	**Added sub-task:** • create spatial indexes
Design data integrity Design table data integrity policy (checks, private key/foreign key, uniqueness, XML schema); select a primary key; data usage patterns	**Added sub-task:** • design a table data integrity policy, including nullability
Design for implicit and explicit transactions Manage transactions; use transactions in code; ensure data integrity by using transactions; use transactions inside the database using T-SQL and from the "outside" via C#/VB; distributed transaction escalation	**Revised task – new full definition:** • manage transactions • ensure data integrity by using transactions • manage distributed transaction escalations • design savepoints • design error handling for transactions, including TRY, CATCH, and THROW

4. Optimize and troubleshoot queries (20-25%)

Tasks Currently Measured	Tasks Added/Changed post *April 2014*
Optimize and tune queries Tune a badly performing query; identify long running queries; review and optimize code; analyze execution plans to optimize queries; tune a query that is poorly written; tune queries using execution plans and database tuning advisor (DTA); design advanced queries: pivots, utilizing common table expressions (CTE), design the database layout and optimize queries (for speed and/or data size); understand different data types; basic knowledge of query hints; tune query workloads, using realistic data sets not being production data sets; demonstrate use of recursive CTE; full text search; control execution plans	**Added sub-tasks:** • implement semantic search • implement plan guides
Troubleshoot and resolve performance problems Interpret performance monitor data; impact of recovery modal on database size, and recovery; how to clean up if .MDF and .LDF files get too large; identify and fix transactional replication problems; detect and resolve server hung, failure; identify and troubleshoot data access problems	**Added sub-tasks:** • integrate performance monitor data with SQL Traces • manage tempdb contention and auto growth • implement Resource Governor • monitor and resolve In-Memory OLTP issues, including merge and garbage collection
Optimize indexing strategies Develop optimal strategy for clustered indexes; analyze index usage; know the difference between the type of indexes and when to choose one over the other; optimize indexing for data warehousing vs. optimize indexing for Online Transaction Processing (OLTP); generate appropriate indexes and statistics with include columns; apply effective and efficient indexes, including the use of INCLUDE lists; full-text indexing	**Added sub-tasks:** • create filtered indexes • implement columnstore indexes • optimize online index maintenance
Capture and analyze execution plans Collect and read execution plan; review an execution plan to spot potential performance issues; read an execution plan; create an index based on an execution plan; row-based logic vs. set-based logic, batching, splitting implicit transactions	**Revised task – new full definition:** • 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	**Revised task – new full definition:** • monitor performance using Dynamic Management Views

Use Data Management Views to determine performance issues; from system metadata; gather trace information by using the SQL Server Profiler; develop monitoring strategy for production database; run a Profiler trace and analyze the results; run Profiler for troubleshooting application; collect output from the Database Engine Tuning Advisor; extended events	• 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