Creation, Compilation, and Invalidation

Kimberly L. Tripp
SQLskills.com
@KimberlyLTripp

pluralsight
hardcore dev and IT training

Overview

- What happens when you create a procedure?
- Where are stored procedures stored?
- Creating stored procedures
- Stored procedure plans and caching
- Side effect: plan cache flush
- Plan invalidation
  - Plan invalidation due to statistics updates
  - Updates to statistics may not invalidate bad plans
- Stored procedure caching
  - Compilation concerns
  - When should you recompile?

What Happens When You Create a Procedure?

- **SQL Server parses the stored procedure to make sure there are no syntactical errors**
- **Object resolution refers to verifying the referenced objects and their dependencies**
  - When ALL of the dependent objects exist
    - All references are verified
    - Finds small errors right away
  - If ANY of the dependent objects do not exist
    - ALL references are deferred until [first] runtime
    - Might not find a small error/typo until testing
- **Once the object is successfully parsed (regardless of resolution), the object is placed into the system tables (the metadata is added)**

Where Are Stored Procedures Stored? (1)

- **Tables and indexes have their own allocations within a database**
- **Stored procedures do not have their own allocations**
  - Their information is stored as rows in internal system tables
  - Information about your stored procedure is accessed through:
    - Catalog views – these are Microsoft-specific views about the metadata
    - Information_schema views – these are ANSI standard views about metadata
- **Accessing metadata about stored procedures within the database**
  - sys.objects
  - sys.procedures
  - sys.sql_modules (is_recompiled)
  - sp_help ['*procedurename*']

Where Are Stored Procedures Stored? (2)

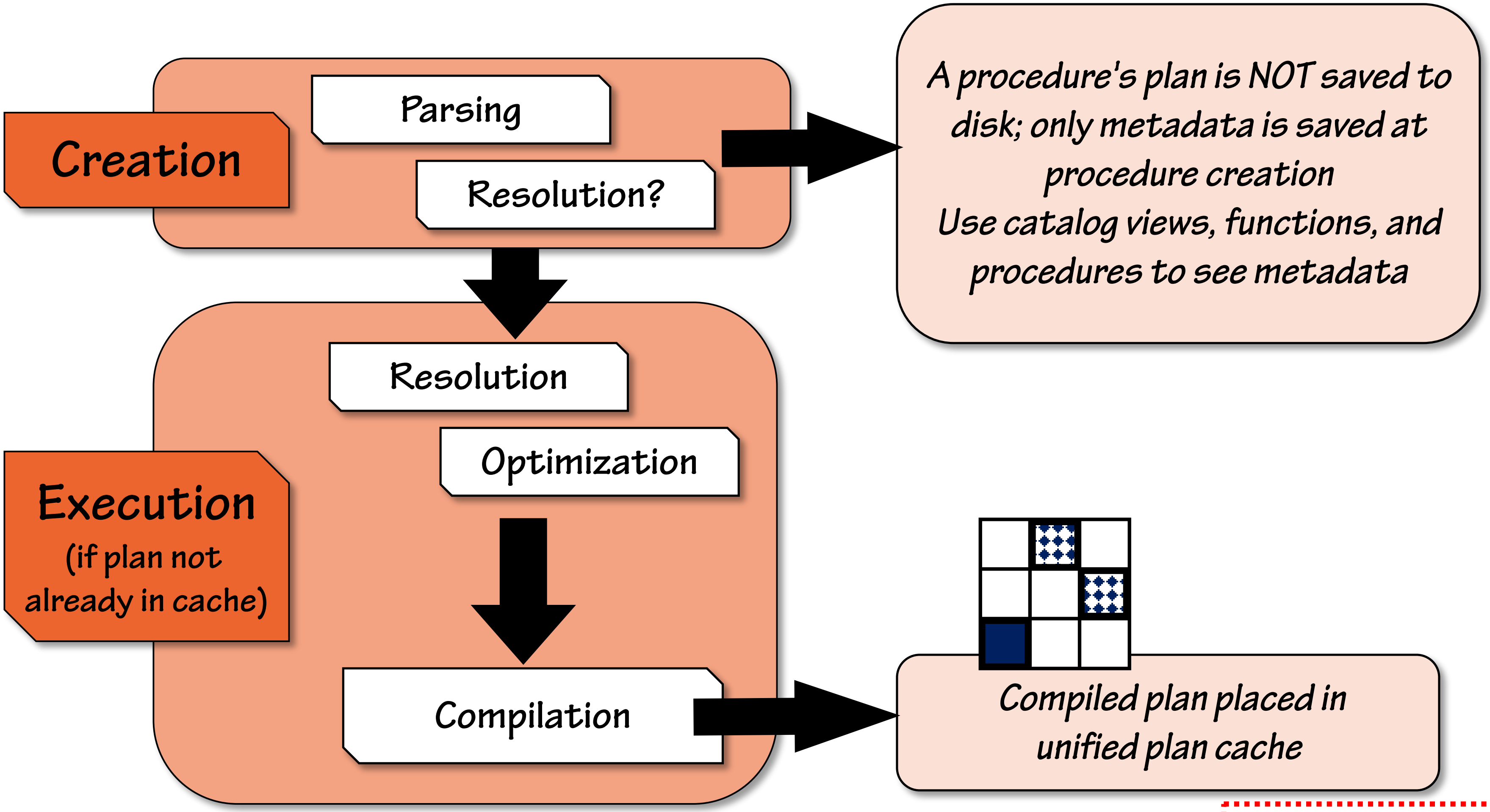
- **Accessing metadata about stored procedures (continued)**
  - Transact-SQL definition of the stored procedure (only if unencrypted):

```
EXEC sp_helptext 'procedurename';

SELECT OBJECT_DEFINITION (OBJECT_ID ('procedurename'));

SELECT OBJECT_NAME([sm].[object_id]) AS [Object Name]
    , [sm].[definition] AS [Object Definition]
FROM [sys].[sql_modules] AS [sm];
```

  - Accessing dependency information:

```
SELECT [ref].*
FROM [sys].[dm_sql_referenced_entities]
    ('procedurename', 'OBJECT') AS [ref];

SELECT [reg].*
FROM [sys].[dm_sql_referencing_entities]
    ('procedurename', 'OBJECT') AS [reg];
```

Creating Stored Procedures



Creation

Parsing

Resolution?

A procedure's plan is NOT saved to disk; only metadata is saved at procedure creation

Use catalog views, functions, and procedures to see metadata

Execution

(if plan not already in cache)

Resolution

Optimization

Compilation

Compiled plan placed in unified plan cache

Stored Procedure Plans and Caching

- **A plan is generated when no plan already exists in cache**
- **Plans are never saved on disk and may not persist within the cache**
  - Some operations completely remove / evict the plan from the cache
    - **Server-level:** Server restart | DBCC FREEPROCCACHE | some RECONFIGURE changes
      - See recordings starting with **Side Effect: Plan Cache Flush** for version-specific details and demo
    - **Database-level:** DBCC FLUSHPROCINDB (undocumented) | sp_dbcmptlevel
    - **Procedure-level:** sp_recompile or they're aged out through non-use
  - Some operations cause the plan to be invalidated (but it still remains an object in the cache); these can be tracked
    - Schema of base object changes (ALTER TABLE / ALTER <object>)
      - Including when indexes are added to the base object
    - Statistics of base objects change
      - See recordings starting with **Plan Invalidation** for version-specific details and demos
- **When a plan exists in cache, all subsequent executions use that plan**

Side Effect: Plan Cache Flush (1)

- **In SQL Server 2005 (only), numerous operations cause the entire plan cache to be cleared / flushed (all plans evicted from cache)**
  - Database-level operations that cause an entire plan cache flush include:
    - When a database auto-closes (from AUTO_CLOSE option)
    - When a database is detached
    - When a database snapshot is dropped
    - When a database state is changed (OFFLINE, ONLINE, READ_ONLY, READ_WRITE)
    - When a database backup is restored
    - …
    - **There are quite a few others;** see KB article 917828 for complete list
  - Server-level operations that cause an entire plan cache flush include:
    - max server memory (MB)
    - min server memory (MB)
    - …
    - **There are quite a few others;** more information coming up

Side Effect: Plan Cache Flush (2)

- **From SQL Server 2005 SP2 onward, messages written to the error log show you that the cache was flushed:**
  - □ SQL Server has encountered 4 occurrence(s) of cachestore flush for the 'Object Plans' cachestore (part of plan cache) due to some database maintenance or reconfigure operations.
  - □ SQL Server has encountered 4 occurrence(s) of cachestore flush for the 'SQL Plans' cachestore (part of plan cache) due to some database maintenance or reconfigure operations.
  - □ SQL Server has encountered 4 occurrence(s) of cachestore flush for the 'Bound Trees' cachestore (part of plan cache) due to some database maintenance or reconfigure operations.

Side Effect: Plan Cache Flush (3)

- **From SQL Server 2008 onward, many database-level operations that caused the entire plan cache to be flushed no longer do so**
  - Instead, they only cause the plans for that database to be flushed

- **Database-level operations that cause the database plan cache to be flushed include:**
  - DBCC FLUSHPROCINDB (dbid)
  - When a database is detached / dropped / restored
  - When a database (or filegroup with the database) changes between READ_ONLY and READ_WRITE
  - When a database changes between OFFLINE and ONLINE, or auto closes
  - There are others… (for example, changing recovery model …)

NOTE: There are many places where a few of these are documented incorrectly. Some aren't detailed at all and others say that many of these operations STILL cause the entire plan cache to be cleared.

Side Effect: Plan Cache Flush (4)

- Many server-level configuration changes cause the entire plan cache to be cleared (at the time of RECONFIGURE, not sp_configure)

- Configuration options that could cause the entire cache to be cleared:

Some Versions	Most Versions	SQL Server 2014
cost threshold for parallelism	cross db ownership chaining	access check cache bucket count
max text repl size (B)	index create memory (KB)	access check cache quota
query governor cost limit	max degree of parallelism	clr enabled
remote query timeout (s)	min *or* max server memory (MB)	max worker threads
	min memory per query (KB)	
	query wait (s)	
	user options	

- Key point: be careful making database-level or server-level configuration changes during production hours
  - Test them in development first
    - sp_configure, then RECONFIGURE – check the cache messages
    - Database changes – check [sys].[dm_exec_procedure_cache]