**Example 20.1**

USE master;

SELECT spid, dbid, uid, cpu

FROM master.dbo.sysprocesses

order by cpu DESC;

**Example 20.2**

SELECT session\_id, database\_id, user\_id, cpu\_time, sql\_handle

FROM sys.dm\_exec\_requests

order by cpu\_time DESC

**Example 20.3**

SELECT TOP 20 SUM(total\_worker\_time) AS cpu\_total,

SUM(execution\_count) AS exec\_ct, COUNT(\*) AS all\_stmts, plan\_handle

FROM sys.dm\_exec\_query\_stats

GROUP BY plan\_handle

ORDER BY cpu\_total;

**Example 20.4**

SELECT type, SUM(pages\_kb)

FROM sys.dm\_os\_memory\_clerks

WHERE pages\_kb != 0

GROUP BY type

ORDER BY 2 DESC;

**Example 20.5**

SELECT type , SUM(pages\_in\_bytes) AS total\_memory

FROM sys.dm\_os\_memory\_objects

GROUP BY type

ORDER BY total\_memory DESC;

Example 20.6

SELECT wait\_type, waiting\_tasks\_count, wait\_time\_ms

FROM sys.dm\_os\_wait\_stats

ORDER BY wait\_type;

**Example 20.7**

SELECT database\_id, file\_id, num\_of\_reads,

num\_of\_bytes\_read, num\_of\_bytes\_written

FROM sys.dm\_io\_virtual\_file\_stats (NULL, NULL);

**Example 20.8**

SELECT net\_transport, auth\_scheme

FROM sys.dm\_exec\_connections

WHERE session\_id=@@SPID;

**Example 20.9**

SELECT num\_reads, num\_writes

FROM sys.dm\_exec\_connections;

**Example 20.10**

USE sample;

CREATE TABLE orders

  (orderid INTEGER NOT NULL,

   orderdate DATE,

   shippeddate DATE,

   freight money);

CREATE TABLE order\_details

  (productid INTEGER NOT NULL,

   orderid INTEGER NOT NULL,

   unitprice money,

   quantity INTEGER);

**Example 20.11**

-- This batch inserts 3000 rows in the table orders

USE sample;

declare @i int, @order\_id integer

           declare @orderdate datetime

           declare @shipped\_date datetime

           declare @freight money

           set @i = 1

           set @orderdate = getdate()

           set @shipped\_date = getdate()

           set @freight = 100.00

          while @i < 3001

           begin

           insert into orders (orderid, orderdate, shippeddate, freight)

             values(@i, @orderdate, @shipped\_date, @freight)

           set @i = @i+1

           end

**Example 20.12**

-- This batch inserts 30000 rows in order\_details and modifies some of them

USE sample

declare @i int, @j int

           set @i = 3000

           set @j = 10

           while @j > 0

           begin

           if @i > 0

               begin

               insert into order\_details (productid, orderid, quantity)

                      values (@i, @j, 5)

              set @i = @i – 1

              end

           else begin

                set @j = @j – 1

                set @i = 3000

               end

           end

go

           update order\_details set quantity = 3

                  where productid in (1511, 2678)

**Example 20.13**

USE sample;

SELECT orders.orderid, orders.shippeddate

        FROM orders

     WHERE orders.orderid between 806 and 1600

     and not exists (SELECT order\_details.orderid

                                  FROM order\_details

                                  WHERE order\_details.orderid = orders.orderid);

**Example 20.14**

CREATE EVENT SESSION session1 ON SERVER

ADD EVENT sqlserver.sql\_batch\_starting(ACTION(package0.event\_sequence,sqlserver.client\_app\_name,  
sqlserver.client\_pid,sqlserver.database\_id,  
sqlserver.database\_name,sqlserver.nt\_username,  
sqlserver.query\_hash,sqlserver.server\_principal\_name,sqlserver.session\_id)

WHERE ([package0].[equal\_boolean]([sqlserver].[is\_system],(0))))

ADD TARGET package0.event\_file(SET filename=N'Session1.xel',max\_file\_size=(5),max\_rollover\_files=(4))

WITH (MAX\_MEMORY=8192 KB,  
EVENT\_RETENTION\_MODE=ALLOW\_SINGLE\_EVENT\_LOSS,

MAX\_DISPATCH\_LATENCY=5 SECONDS,MAX\_EVENT\_SIZE=0 KB,   
 MEMORY\_PARTITION\_MODE=PER\_CPU,  
TRACK\_CAUSALITY=ON,STARTUP\_STATE=OFF)

**Example 20.15**

USE master;

ALTER EVENT SESSION Session1 ON SERVER STATE=start

**Example 20.16**

USE master;

ALTER EVENT SESSION Session1 ON SERVER STATE=stop

**Example 20.17**

USE master;

SELECT sessions.name AS SessionName, event1.package as PackageName,

event1.name AS EventName,action1.name AS ActionName

FROM sys.server\_event\_sessions sessions

INNER JOIN sys.server\_event\_session\_events event1

ON sessions.event\_session\_id = event1.event\_session\_id

INNER JOIN sys.server\_event\_session\_actions action1

ON sessions.event\_session\_id = action1.event\_session\_id

WHERE sessions.name = 'session1' ;

**Example 20.18**

USE master;

SELECT pkg.name as PackageName, obj.name as EventName

FROM sys.dm\_xe\_packages pkg

INNER JOIN sys.dm\_xe\_objects obj on pkg.guid = obj.package\_guid

WHERE obj.object\_type = 'event' AND pkg.name = 'filestream'

ORDER by 2;

**Example 20.19**

USE master;

ALTER DATABASE AdventureWorks SET QUERY\_STORE=ON;

**Example 20.20**

USE master;

GO

ALTER DATABASE AdventureWorks

SET QUERY\_STORE ( OPERATION\_MODE = READ\_WRITE,

CLEANUP\_POLICY = (STALE\_QUERY\_THRESHOLD\_DAYS = 30),

DATA\_FLUSH\_INTERVAL\_SECONDS = 3000, MAX\_STORAGE\_SIZE\_MB = 500,

INTERVAL\_LENGTH\_MINUTES = 15, QUERY\_CAPTURE\_MODE = AUTO,

MAX\_PLANS\_PER\_QUERY = 1000);

**Example 20.21**

USE AdventureWorks;

SELECT q.query\_id, qt.query\_text\_id, qt.query\_sql\_text,

SUM(rs.count\_executions) AS execution\_count

FROM sys.query\_store\_query\_text AS qt

JOIN sys.query\_store\_query AS q ON qt.query\_text\_id = q.query\_text\_id

JOIN sys.query\_store\_plan AS p ON q.query\_id = p.query\_id

JOIN sys.query\_store\_runtime\_stats AS rs ON p.plan\_id = rs.plan\_id

GROUP BY q.query\_id, qt.query\_text\_id, qt.query\_sql\_text

ORDER BY execution\_count DESC;

**Example 20.22**

USE AdventureWorks;

SELECT TOP 10 rs.avg\_duration, qt.query\_sql\_text, q.query\_id,

qt.query\_text\_id, p.plan\_id, GETUTCDATE() AS curr\_time,

rs.last\_execution\_time

FROM sys.query\_store\_query\_text AS qt

JOIN sys.query\_store\_query AS q

ON qt.query\_text\_id = q.query\_text\_id

JOIN sys.query\_store\_plan AS p

ON q.query\_id = p.query\_id

JOIN sys.query\_store\_runtime\_stats AS rs

ON p.plan\_id = rs.plan\_id

WHERE rs.last\_execution\_time > DATEADD(HOUR, -1, GETUTCDATE())

ORDER BY rs.avg\_duration DESC;

**Example 20.23**

-- Copy Sales.OrderLines and Sales.Orders in the sample DB

USE sample;

SELECT \* INTO salesorderLines

FROM WideWorldImporters.Sales.OrderLines;

SELECT \* INTO salesorders

FROM WideWorldImporters.Sales.Orders;

**Example 20.24**

USE master;

GO

-- Enable Query Store

ALTER DATABASE sample SET QUERY\_STORE = ON;

GO

-- Configure Query Store

ALTER DATABASE sample SET QUERY\_STORE

(OPERATION\_MODE = READ\_WRITE,

DATA\_FLUSH\_INTERVAL\_SECONDS = 600,

MAX\_STORAGE\_SIZE\_MB = 500,

INTERVAL\_LENGTH\_MINUTES = 30);

**Example 20.25**

-- Clear procedure cache

USE sample;

GO

-- Clear procedure cache

DBCC FREEPROCCACHE;

GO

-- Clear Query Store

ALTER DATABASE CURRENT SET QUERY\_STORE CLEAR ALL;

**Example 20.26**

EXEC sp\_executesql N'SELECT SUM(UnitPrice\*Quantity)

FROM SalesOrderLines sl

JOIN salesorders so ON sl.OrderID=so.OrderID

WHERE PackageTypeID = @typeID', N'@typeID int', @typeID = 7;

GO 600

**Example 20.27**

USE sample;

DBCC FREEPROCCACHE;

EXEC sp\_executesql N'SELECT SUM(UnitPrice\*Quantity)

FROM SalesOrderLines sl JOIN salesorders so ON sl.OrderID=SO.OrderID

WHERE PackageTypeID = @typeID', N'@typeID INT', @typeID = 0;

**Example 20.28**

EXEC sp\_executesql N'SELECT SUM(UnitPrice\*Quantity)

FROM SalesOrderLines sl JOIN salesorders so ON sl.OrderID=so.OrderID

where PackageTypeID = @typeID', N'@typeID INT', @typeID = 7;

GO 20

**Example 20.29**

USE sample;

SELECT reason, score,

script=JSON\_VALUE(details, '$.implementationDetails.script')

FROM sys.dm\_db\_tuning\_recommendations

CROSS APPLY OPENJSON (Details, '$.planForceDetails')

WITH ([query\_id] int '$.queryId',

regressedPlanId int '$.regressedPlanId',

recommendedPlanId int '$.recommendedPlanId',

regressedPlanErrorCount int,

recommendedPlanErrorCount int,

regressedPlanExecutionCount int,

regressedPlanCpuTimeAverage float,

recommendedPlanExecutionCount int,

recommendedPlanCpuTimeAverage float

) AS planForceDetails;

**Example 20.30**

USE master;

ALTER DATABASE sample

SET AUTOMATIC\_TUNING (FORCE\_LAST\_GOOD\_PLAN = ON );

**Example 20.31**

USE sample;

SELECT name, state

FROM sys.dm\_db\_tuning\_recommendations

CROSS APPLY OPENJSON (Details, '$.planForceDetails')

WITH ([query\_id] int '$.queryId',

regressedPlanId int '$.regressedPlanId',

recommendedPlanId int '$.recommendedPlanId',

regressedPlanErrorCount int,

recommendedPlanErrorCount int,

regressedPlanExecutionCount int,

regressedPlanCpuTimeAverage float,

recommendedPlanExecutionCount int,

recommendedPlanCpuTimeAverage float

) AS planForceDetails;

**Example 20.32**

USE sample;

SELECT name, actual\_state, desired\_state, reason

FROM sys.database\_automatic\_tuning\_options;