## The SELECT statement

```
SELECT CountryRegion, COUNT(*) as NumberOfRows
FROM [SalesLT].[Address]
WHERE ModifiedDate < '2010-01-01'
GROUP BY CountryRegion
HAVING COUNT(*)>100
ORDER BY CountryRegion DESC
```

## JOINing two tables together

```
SELECT H.SalesOrderID, OrderDate, OrderQty
FROM [SalesLT].[SalesOrderDetail] AS D
JOIN [SalesLT].[SalesOrderHeader] AS H
ON D.SalesOrderID = H.SalesOrderID
```

## Data Types

```
SELECT N'Hi'
```

## 58. Determine the appropriate type of execution plan

```
SET SHOWPLAN_TEXT OFF
GO

SELECT *
FROM SalesLT.Address A
CROSS JOIN SalesLT.Address B
```

## 59. Loops and Scans

```
SELECT * FROM SalesLT.Address
where City = 'Washington'

CREATE NONCLUSTERED INDEX [IX_Address_City] ON [SalesLT].[Address]
(
[City] ASC
)
GO

SELECT *
FROM SalesLT.Address A
CROSS JOIN SalesLT.Address B


SELECT H.CustomerID, H.SalesOrderID, D.OrderQty
FROM SalesLT.SalesOrderHeader H
INNER JOIN SalesLT.SalesOrderDetail D
ON H.SalesOrderID = D.SalesOrderID

SELECT *
INTO [SalesLT].[SalesOrderDetailCopy]
from [SalesLT].[SalesOrderDetail]

SELECT *
INTO [SalesLT].[SalesOrderHeaderCopy]
from [SalesLT].[SalesOrderHeader]

SELECT H.CustomerID, H.SalesOrderID, D.OrderQty
FROM SalesLT.SalesOrderHeaderCopy H
INNER JOIN SalesLT.SalesOrderDetailCopy D
```

```
ON H.SalesOrderID = D.SalesOrderID

SELECT COUNT(*)
FROM [SalesLT].[SalesOrderHeaderCopy]
INSERT INTO [SalesLT].[SalesOrderHeaderCopy]
SELECT H1.*
FROM [SalesLT].[SalesOrderHeaderCopy] as H1
```

## 59. Problems in execution plans

```
SELECT City FROM SalesLT.Address
where YEAR(ModifiedDate) = 2006 -- Not SARGable
CREATE NONCLUSTERED INDEX IX_Address_Modified ON SalesLT.Address(ModifiedDate,City)
SELECT City FROM SalesLT.Address
WHERE ModifiedDate BETWEEN '2006-01-01' and '2006-12-31 23:59:59' –SARGable

create proc NameOfProc (@Year int) WITH RECOMPILE as
SELECT * FROM SalesLT.Address
WHERE ModifiedDate BETWEEN '2006-01-01' and '2006-12-31 23:59:59' --SARGable
order by ModifiedDate
OPTION (RECOMPILE)

SELECT * FROM SalesLT.Address
WHERE LEFT(AddressLine1,1) = '8' -- Not SARGable

SELECT * FROM SalesLT.Address
WHERE AddressLine1 LIKE '8%' – SARGable

SELECT LEN(AddressLine1)
FROM SalesLT.Address
order by LEN(AddressLine1)
```

## 63. Index changes

```
CREATE NONCLUSTERED INDEX ix_Address_AddressLine1_AddressLine2
ON [SalesLT].[Address](AddressLine1, AddressLine2)
WHERE [City] = 'Bothell'
WITH (FILLFACTOR = 62)
```

## 64. Finding missing index

```
SELECT H.CustomerID, H.SalesOrderID, D.OrderQty
FROM SalesLT.SalesOrderHeaderCopy H
INNER JOIN SalesLT.SalesOrderDetailCopy D
ON H.SalesOrderID = D.SalesOrderID

select count(*) from [SalesLT].[SalesOrderHeaderCopy]

insert into [SalesLT].[SalesOrderHeaderCopy]
select H1.*
from [SalesLT].[SalesOrderHeaderCopy] as H1

select * from sys.dm_db_missing_index_details

SELECT
   CONVERT (varchar, getdate(), 126) AS runtime
   , mig.index_group_handle
   , mid.index_handle
   , CONVERT (decimal (28,1), migs.avg_total_user_cost * migs.avg_user_impact *
        (migs.user_seeks + migs.user_scans)) AS improvement_measure
```

```
   , 'CREATE INDEX missing_index_' + CONVERT (varchar, mig.index_group_handle) + '_' +
        CONVERT (varchar, mid.index_handle) + ' ON ' + mid.statement + '
        (' + ISNULL (mid.equality_columns,'')
        + CASE WHEN mid.equality_columns IS NOT NULL
        AND mid.inequality_columns IS NOT NULL
        THEN ',' ELSE '' END + ISNULL (mid.inequality_columns, '') + ')'
        + ISNULL (' INCLUDE (' + mid.included_columns + ')', '') AS
create_index_statement
   , migs.*
   , mid.database_id
   , mid.[object_id]
FROM sys.dm_db_missing_index_groups AS mig
   INNER JOIN sys.dm_db_missing_index_group_stats AS migs
      ON migs.group_handle = mig.index_group_handle
   INNER JOIN sys.dm_db_missing_index_details AS mid
      ON mig.index_handle = mid.index_handle
 ORDER BY migs.avg_total_user_cost * migs.avg_user_impact * (migs.user_seeks +
migs.user_scans) DESC
```

## 65. Assess the use of hints for query performance

```
SELECT H.CustomerID, H.SalesOrderID, D.OrderQty
FROM SalesLT.SalesOrderHeaderCopy H
INNER JOIN SalesLT.SalesOrderDetailCopy D
ON H.SalesOrderID = D.SalesOrderID
OPTION (LOOP JOIN)
```

## 24. Create users from Azure AD identities

```
CREATE USER [Susan@Filecats.onmicrosoft.com]
FROM EXTERNAL PROVIDER
```

## 25, 113. Configure security principals

```
ALTER ROLE [db_owner]
ADD MEMBER [Susan@Filecats.onmicrosoft.com]

REVOKE SELECT ON OBJECT::[SalesLT].[Address]
TO [Susan@Filecats.onmicrosoft.com]

CREATE PROC SalesLT.StoredProcedure WITH EXECUTE AS [Susan@Filecats.onmicrosoft.com] AS
...

SELECT * FROM sys.fn_my_permissions(NULL, 'DATABASE')
SELECT * FROM sys.fn_my_permissions('DP300', 'DATABASE')
SELECT * FROM sys.fn_my_permissions('SalesLT.Customer', 'OBJECT')
```

## 114. Custom roles

```
CREATE ROLE MyCustomRole1

GRANT SELECT ON OBJECT::[SalesLT].[Address]
TO MyCustomRole1

ALTER ROLE MyCustomRole1
ADD MEMBER [Susan@Filecats.onmicrosoft.com]

--

CREATE ROLE MyCustomRole2
```

```
GRANT SELECT ON OBJECT::[SalesLT].[Customer]
TO MyCustomRole2

ALTER ROLE MyCustomRole2
ADD MEMBER [Susan@Filecats.onmicrosoft.com]

--

CREATE ROLE MyCustomRole3

REVOKE SELECT ON OBJECT::[SalesLT].[Customer]
TO MyCustomRole2

ALTER ROLE MyCustomRole3
ADD MEMBER [Susan@Filecats.onmicrosoft.com]

--

GRANT SELECT ON OBJECT::[SalesLT].[Customer]
TO [public]
```

## 28. Implement Transparent Data Encryption (TDE)

```
ALTER DATABASE DP300 SET ENCRYPTION ON
```

## 29, 33. Implement Always Encrypted

```
declare @myCity as nvarchar(30) = 'Bothell'
select *
from [SalesLT].[Address]
where city = @myCity
```

## 32. Configure server and database-level firewall rules

```
SELECT * FROM sys.firewall_rules

EXECUTE sp_set_firewall_rule @name = N'MyFirewallRule', @start_ip_address =
'86.134.144.143', @end_ip_address = '86.134.144.144'

EXECUTE sp_delete_firewall_rule @name = N'MyFirewallRule'

-- Database-level IP Firewall rules

SELECT * FROM sys.database_firewall_rules

EXECUTE sp_set_database_firewall_rule @name = N'MyDatabaseFirewallRule',
@start_ip_address = '192.168.1.1', @end_ip_address = '192.168.1.200'

EXECUTE sp_delete_database_firewall_rule @name = N'MyDatabaseFirewallRule'
```

## 34. Apply a data classification strategy

```
SELECT * FROM sys.sensitivity_classifications

select * from sys.columns where object_id = 1506104406

ADD SENSITIVITY CLASSIFICATION TO
[SalesLT].[Address].PostalCode
WITH (
LABEL='Highly Confidential',
```

```
INFORMATION_TYPE='Financial',
RANK=LOW
)

DROP SENSITIVITY CLASSIFICATION FROM [SalesLT].[Address].PostalCode
```

## 36. Implement Data Change Tracking – CT

```
-- Check which tables/databases have CT enabled:

SELECT * from sys.change_tracking_databases
select * from sys.databases
SELECT * from sys.change_tracking_tables
select * from sys.objects

-- Get the initial sync version

DECLARE @last_sync bigint;
SET @last_sync = CHANGE_TRACKING_CURRENT_VERSION();
select @last_sync
select CHANGE_TRACKING_CURRENT_VERSION()

-- After changes have happened:

SELECT  CT.AddressID, CT.SYS_CHANGE_OPERATION,
CT.SYS_CHANGE_COLUMNS, CT.SYS_CHANGE_CONTEXT
FROM  CHANGETABLE(CHANGES SalesLT.Address, 0) AS CT

-- CHANGE_TRACKING_IS_COLUMN_IN_MASK

-- Check that you don't have to refresh the entire table:
SELECT @last_sync >= CHANGE_TRACKING_MIN_VALID_VERSION(OBJECT_ID('SalesLT.Address'))
--     80            90
```

## 36. Implement Data Change Tracking - CDC

```
-- Enable for database

EXEC sys.sp_cdc_enable_db

-- Enable for table

EXEC sys.sp_cdc_enable_table @source_schema = 'SalesLT', @source_name = 'Address',
                                                @role_name = 'NewRole',
@captured_column_list = 'AddressID, City'

-- What is my configuration?
EXECUTE sys.sp_cdc_help_change_data_capture

-- Update row

SELECT * FROM [SalesLT].[Address] WHERE AddressID = 9
UPDATE [SalesLT].[Address]
SET City = 'Bothell'
WHERE AddressID = 9

-- What are the changed rows?
```

```
  DECLARE @from_lsn binary(10), @to_lsn binary(10);
SET @from_lsn = sys.fn_cdc_get_min_lsn('SalesLT_Address');
SET @to_lsn = sys.fn_cdc_get_max_lsn();
SELECT * FROM cdc.fn_cdc_get_all_changes_SalesLT_Address  (@from_lsn, @to_lsn, N'all');

-- Disable from table

EXEC sys.sp_cdc_disable_table @source_schema = 'SalesLT', @source_name = 'Address',
                                                    @capture_instance = 'all'

-- Disable for database

EXEC sys.sp_cdc_disable_db
```

## 42. Configure and monitor activity and performance

```
EXEC sp_who -- current users/processes
EXEC sp_lock
EXEC sp_spaceused
EXEC sp_monitor – statistics
```

## 43, 50. Implement Index Maintenance tasks

```
-- Find missing indexes

SELECT * FROM sys.dm_db_missing_index_details

-- Assess fragmentation of database indexes

SELECT db_name(database_id) as DBName, object_name(object_id) as ObjectName,
avg_fragmentation_in_percent, page_count
FROM sys.dm_db_index_physical_stats(NULL,NULL,NULL,NULL,NULL)
ORDER BY avg_fragmentation_in_percent * page_count desc

DBCC SHOWCONTIG

-- Assess columnstore indexes

SELECT deleted_rows, total_rows
FROM sys.dm_db_column_store_row_group_physical_stats

-- Rebuild/Reorganize indexes

ALTER INDEX ALL
ON [SalesLT].[Address]
REORGANIZE

ALTER INDEX [PK_Customer_CustomerID]
ON [SalesLT].[Customer]
REBUILD WITH (ONLINE = ON,
              FILLFACTOR = 70,
                      MAX_DURATION = 30,
                      RESUMABLE = ON)

ALTER INDEX [PK_Customer_CustomerID]
ON [SalesLT].[Customer]
PAUSE -- or ABORT or RESUME
```

## 44. Implement statistics maintenance tasks

```
-- Update all user-defined and internal tables

EXEC sp_updatestats

-- Update a particular table or indexed view

UPDATE STATISTICS [SalesLT].[Address] [AK_Address_rowguid]
WITH FULLSCAN
--WITH SAMPLE 10 PERCENT
--WITH RESAMPLE
--, PERSIST_SAMPLE_PERCENT = ON
```

## 45. Configure database auto-tuning

```
-- Which indexes are auto-created?

SELECT * FROM sys.indexes
WHERE auto_created = 1

-- Change database auto-tuning

ALTER DATABASE [DP300] SET AUTOMATIC_TUNING = AUTO --| INHERIT | CUSTOM
ALTER DATABASE [DP300] SET AUTOMATIC_TUNING (FORCE_LAST_GOOD_PLAN = ON, CREATE_INDEX =
ON, DROP_INDEX = OFF)

-- What are the tuning recommendations?

SELECT * FROM sys.dm_db_tuning_recommendations
```

## 47. Manage storage capacity

```
-- Display allocated space

   -- For a single database. Needs to be in the master database.
SELECT database_name, allocated_storage_in_megabytes FROM sys.resource_stats -- Single
database
   -- For an elastic pool. Needs to be in the master database.
SELECT elastic_pool_name, elastic_pool_storage_limit_mb, avg_allocated_storage_percent
FROM sys.elastic_pool_resource_stats -- elastic pool

--Display maximum size (not "Master" database):

SELECT DATABASEPROPERTYEX('DP300', 'MaxSizeInBytes')

   -- View current log size
SELECT file_id, type_desc, size, max_size, growth
FROM sys.database_files
WHERE type = 1

-- To shrink a transaction log file

DBCC SHRINKFILE(2) -- where 2 = file_id from above
DBCC SHRINKDATABASE(DP300)
```

## 50, 105. Assess growth of databases

```
-- Assess growth in a database
   -- For a database (need to be in the "Master" database)
SELECT database_name, start_time, storage_in_megabytes
```

```
FROM sys.resource_stats
ORDER BY database_name, start_time

   -- For an elastic pool (need to be in the "Master" database)
SELECT start_time, elastic_pool_name, elastic_pool_storage_limit_mb,
avg_allocated_storage_percent
FROM sys.elastic_pool_resource_stats
ORDER BY start_time

-- Report on database free space

EXEC sp_spaceused

-- Display by file

SELECT file_id, name, type_desc, physical_name, size, max_size
FROM sys.database_files

-- View number of pages

SELECT allocated_extent_page_count, unallocated_extent_page_count
FROM sys.dm_db_file_space_usage

DBCC SQLPERF (LOGSPACE)

-- View tempdb database size

SELECT * FROM sys.dm_db_session_space_usage

SELECT * FROM sys.dm_db_task_space_usage
```

## 48. Configure Query Store to collect performance data

```
-- Enable Query Store

ALTER DATABASE [DP300]
SET QUERY_STORE = ON
(OPERATION_MODE = READ_WRITE)
-- WAIT_STATS_CAPTURE_MODE = ON
-- MAX_STORAGE_SIZE_MB = 500
-- DATA_FLUSH_INTERVAL_SECONDS = 3000
-- SIZE_BASED_CLEANUP_MODE = AUTO
-- CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = 30)
-- INTERVAL_LENGTH_MINUTES = 15
-- QUERY_CAPTURE_MODE = AUTO
-- MAX_PLANS_PER_QUERY = 1000
-- WAIT_STATS_CAPTURE_MODE = ON

-- View query store options

select * from sys.database_query_store_options

-- To clear Query Store

ALTER DATABASE [DP300] SET QUERY_STORE CLEAR;

-- Query plans
```

```
SELECT * FROM sys.query_store_plan

-- Query plan statistics

SELECT * FROM sys.query_store_runtime_stats

-- See the queries

SELECT Txt.query_text_id, Txt.query_sql_text, Qry.*
FROM sys.query_store_query AS Qry
INNER JOIN sys.query_store_query_text AS Txt
    ON Qry.query_text_id = Txt.query_text_id ;
```

## 49. Identify sessions that cause blocking

```
-- Session 1

BEGIN TRANSACTION
UPDATE [SalesLT].[Address]
SET City = 'Toronto ON'
where City = 'Toronto'

-- Session 2

BEGIN TRANSACTION
UPDATE [SalesLT].[Address]
SET City = 'Toronto'
where City in ('Toronto ON', 'Toronto')

--To view locks:

SELECT * FROM sys.dm_tran_locks

--To view blocking:

SELECT session_id, blocking_session_id,
    start_time, status, command,
    DB_NAME(database_id) as [database],
    wait_type, wait_resource, wait_time,
    open_transaction_count
FROM sys.dm_exec_requests
WHERE blocking_session_id > 0;
```

## 49. Serialization level

```
DBCC USEROPTIONS

SET TRANSACTION ISOLATION LEVEL
READ UNCOMMITTED
-- READ COMMITTED
-- REPEATABLE READ
-- SNAPSHOT
-- SERIALIZABLE

ALTER DATABASE [DP300]
SET ALLOW_SNAPSHOT_ISOLATION ON

ALTER DATABASE [DP300]
SET READ_COMMITTED_SNAPSHOT ON
```

## 51, 55. Database configurations

```
ALTER DATABASE [DP300]
SET AUTO_CREATE_STATISTICS OFF
GO

ALTER DATABASE [DP300]
SET AUTO_SHRINK ON
GO

ALTER DATABASE SCOPED CONFIGURATION
-- [FOR SECONDARY]
SET GLOBAL_TEMPORARY_TABLE_AUTO_DROP = ON
-- LAST_QUERY_PLAN_STATS
-- LEGACY_CARDINALITY_ESTIMATION
-- MAXDOP
-- OPTIMIZE_FOR_AD_HOC_WORKLOADS
-- PARAMETER_SNIFFING
-- QUERY_OPTIMIZER_HOTFIXES
GO
```

## 57. Configure Intelligent Query Processing (IQP)

```
-- Server-wide configuration options

SELECT * FROM sys.configurations
EXEC sp_configure '101', 0 -- Not in Azure SQL Database

-- Database-wide configuration options

SELECT * FROM sys.database_scoped_configurations

ALTER DATABASE SCOPED CONFIGURATION
SET [BATCH_MODE_ON_ROWSTORE] = ON

SELECT * FROM [SalesLT].[Address]
OPTION (USE HINT ('DISALLOW_BATCH_MODE'))
```

## 46. Automate database maintenance tasks

```
-- Create Credentials in the Job database

CREATE MASTER KEY ENCRYPTION BY PASSWORD='<an6?%9++Vyd%Ut9';
GO
CREATE DATABASE SCOPED CREDENTIAL [MasterCred] WITH IDENTITY = 'MasterU', SECRET =
'<an6?%9++Vyd%Ut9'
GO
CREATE DATABASE SCOPED CREDENTIAL [RunJob] WITH IDENTITY = 'JobU', SECRET =
'<an6?%9++Vyd%Ut9'

-- Create a target group

EXEC jobs.sp_add_target_group 'GrpDatabase';

-- Create a target group member

EXEC jobs.sp_add_target_group_member
@target_group_name = 'GrpDatabase',
@target_type = 'SqlDatabase',
-- or 'SqlServer', -- or 'PoolGroup'
```

```
-- if wanting to exclude, @membership_type = 'Exclude'
-- If targeting a server or pool, @refresh_credential_name = 'RefreshPassword',
@server_name = 'dp300database.database.windows.net',
@database_name = 'dp300';

SELECT * FROM jobs.target_groups
WHERE target_group_name='GrpDatabase';

SELECT * FROM jobs.target_group_members
WHERE target_group_name='GrpDatabase';

-- In Master Database

CREATE LOGIN MasterU WITH PASSWORD ='<an6?%9++Vyd%Ut9'
CREATE USER MasterU FROM LOGIN MasterU
CREATE LOGIN JobU WITH PASSWORD = '<an6?%9++Vyd%Ut9'

-- In Target User Database

CREATE USER JobU FROM LOGIN JobU
ALTER ROLE db_owner ADD MEMBER JobU

-- Create a job and job steps

EXEC jobs.sp_add_job @job_name='My first job', @description='Look at objects'

EXEC jobs.sp_add_jobstep @job_name='My first job',
@command='SELECT * FROM sys.objects',
@credential_name='RunJob',
@target_group_name='GrpDatabase'

-- Run/schedule the job in T-SQL

EXEC jobs.sp_start_job 'My first job' -- run now
EXEC jobs.sp_update_job
@job_name='My first job',
@enabled=1,
@schedule_interval_type='Minutes', -- Or Hours, Days, Weeks, Months or Once,
@schedule_interval_count=1

-- Monitor job execution

SELECT * FROM jobs.job_executions
order by start_time
```

## 101. Evaluate database health using DMVs

```
-- CPU, IO and memory

SELECT * from sys.dm_db_resource_stats

-- Storage in the current database or elastic pool

SELECT * from sys.dm_user_db_resource_governance

-- CPU, memory and I/O resource at the SQL Server level.

SELECT * FROM sys.dm_os_job_object
```

```
-- I/O statistics for data and log files

SELECT * FROM sys.dm_io_virtual_file_stats(null, null)

-- Performance counter information

SELECT * FROM sys.dm_os_performance_counters

-- Session 1

BEGIN TRANSACTION
UPDATE [SalesLT].[Address]
SET City = 'Toronto ON'
where City = 'Toronto'

-- Session 2

BEGIN TRANSACTION
UPDATE [SalesLT].[Address]
SET City = 'Toronto'
where City in ('Toronto ON', 'Toronto')

-- Waiting on resources

SELECT * FROM sys.dm_os_wait_stats
SELECT * FROM sys.dm_db_wait_stats

-- Possible blocking

SELECT * FROM sys.dm_exec_requests
WHERE blocking_session_id <> 0

SELECT * FROM sys.dm_os_waiting_tasks
WHERE blocking_session_id <> 0
```

## 102. Evaluate server health

```
SELECT * from sys.databases
SELECT * from sys.objects
SELECT * FROM sys.dm_os_schedulers where STATUS = 'VISIBLE ONLINE';
SELECT SERVERPROPERTY('EngineEdition');

SELECT * FROM sys.resource_usage
```

## 103. Perform database consistency checks by using DBCC

```
-- Check logical and physical integrity of all objects

DBCC CHECKDB
DBCC CHECKDB(DP300)

-- Checks the consistency of disk space allocation structures

DBCC CHECKALLOC
DBCC CHECKALLOC(DP300)

-- Checks all tables and index views.

DBCC CHECKTABLE('[SalesLT].[Address]')
```

```
-- Checks

DBCC CHECKCATALOG
DBCC CHECKCATALOG(DP300)

-- DBCC CHECKDB, CHECKALLOC, CHECKTABLE and CHECKCATALOG options:

DBCC CHECKDB (0, NOINDEX)
ALTER DATABASE [DP300]
SET SINGLE_USER WITH ROLLBACK IMMEDIATE
GO

DBCC CHECKDB (0, REPAIR_REBUILD)
GO
ALTER DATABASE [DP300]
SET MULTI_USER
GO
DBCC CHECKDB (0, REPAIR_FAST)

ALTER DATABASE [DP300]
SET EMERGENCY
GO

ALTER DATABASE [DP300]
SET SINGLE_USER WITH ROLLBACK IMMEDIATE
GO
DBCC CHECKDB (0, REPAIR_ALLOW_DATA_LOSS)

-- DBCC CHECKCATALOG WITH (optional)

WITH NO_INFOMSGS

-- DBCC CHECKDB, CHECKALLOC and CHECKTABLE optional options:

DBCC CHECKDB (0, REPAIR_ALLOW_DATA_LOSS)
WITH
  ALL_ERRORMSGS
, EXTENDED_LOGICAL_CHECKS -- not ALLOC
, NO_INFOMSGS
, TABLOCK
, ESTIMATEONLY
, PHYSICAL_ONLY -- not ALLOC
, MAXDOP = 4

DBCC CHECKCONSTRAINTS
```

## 106. Review database configuration options

```
ALTER DATABASE [DP300]
SET AUTO_CLOSE ON

ALTER DATABASE [DP300]
SET AUTO_CREATE_STATISTICS ON

ALTER DATABASE [DP300]
SET AUTO_UPDATE_STATISTICS ON -- [_ASYNC]
```

```
ALTER DATABASE [DP300]
SET AUTO_SHRINK OFF --ON

ALTER DATABASE [DP300]
SET READ_WRITE --/ READ_ONLY

ALTER DATABASE [DP300]
SET MULTI_USER --/ RESTRICTED_USER / SINGLE_USER

ALTER DATABASE [DP300]
SET RECOVERY FULL --/ RECOVERY BULK_LOGGED / RECOVERY SIMPLE

ALTER DATABASE [DP300]
SET COMPATIBILITY_LEVEL = 150 --(SQL Server 2008 and R2), 110, 120, 130, 140, 150 (SQL
Server 2019)
```

## 10 and 11. Data for Azure SQL MI and Azure VMs

```
create table myTable
(objectname varchar(60),
IDmyTable int primary key identity(1,1))

insert into myTable(objectname)
select A.[name]
from sys.objects as A
cross join sys.objects as B

select * from myTable
```

## 24. Create users from Azure AD identities

```
CREATE LOGIN MyLogin
WITH PASSWORD = 'MyComplexPassword';

CREATE USER MyLogin FOR LOGIN MyLogin;


CREATE LOGIN [Susan@Filecats.onmicrosoft.com]
FROM EXTERNAL PROVIDER
-- DEFAULT_DATABASE =
-- DEFAULT_LANGUAGE =

CREATE USER [Susan@Filecats.onmicrosoft.com]
FOR LOGIN [Susan@Filecats.onmicrosoft.com]


SELECT * FROM sys.server_principals
```

## 111. Manage certificates

```
-- Create a self-signed certificate

CREATE CERTIFICATE CertificateName
ENCRYPTION BY PASSWORD = 'ComplicatedPassw0rd!'
WITH SUBJECT = 'CertificateSubjectName',
     EXPIRY_DATE = '20291231';
GO

DROP CERTIFICATE CertificateName
GO
```

```
-- Creates certificate with existing private key

CREATE CERTIFICATE CertificateName2
    FROM FILE = 'C:\Certs\Certificate.cer'
    WITH PRIVATE KEY (FILE = 'C:\Certs\Certificate\PrivateKey.pvk',
    DECRYPTION BY PASSWORD = 'ComplicatedPassw0rd!2');
GO

-- Alters/removes/imports private key

ALTER CERTIFICATE CertificateName2
--REMOVE PRIVATE KEY
WITH PRIVATE KEY (FILE = 'C:\Certs\Certificate\PrivateKey2.pvk',
    DECRYPTION BY PASSWORD = 'ComplicatedPassw0rd!3');

-- Creates certificate from assembly (DLL file)

CREATE CERTIFICATE CertificateName3
    FROM EXECUTABLE FILE = 'C:\Certs\Certificate\Assembly.dll';
GO
```

## 25. Configure security principals (MI and VM)

```
SELECT * FROM sys.server_principals
SELECT * FROM sys.sql_logins
SELECT * FROM sys.login_token

ALTER SERVER ROLE [diskadmin]
ADD MEMBER [Susan@Filecats.onmicrosoft.com]

EXEC sp_helprotect
EXEC sp_helprole
EXEC sp_helprolemember

CREATE SERVER ROLE newrole AUTHORIZATION [Jane@Filecats.onmicrosoft.com]
GO
ALTER SERVER ROLE newrole
ADD MEMBER [Susan@Filecats.onmicrosoft.com]
GO
USE master
GO
GRANT ALTER ON LOGIN::[MyLogin] TO [newrole]
GO
```

## 70. Recommend table and index storage including filegroups

```
ALTER DATABASE [dp300mia]
ADD FILEGROUP [NewFileGroup]
GO

ALTER DATABASE [dp300mia]
ADD FILE (NAME = N'NewData2',
-- FILENAME = N'C:\PathToData\NewData.ndf' , -- this is for VMs
SIZE = 8192KB , FILEGROWTH = 65536KB )
-- or FILEGROWTH = 10%
TO FILEGROUP [NewFileGroup]
GO

ALTER DATABASE [dp300mia]
```

```
MODIFY FILEGROUP [NewFileGroup]
AUTOGROW_ALL_FILES

-- Add a second filegroup

ALTER DATABASE [dp300mia]
ADD FILEGROUP [NewFileGroup2]
GO

ALTER DATABASE [dp300mia]
ADD FILE (NAME = N'NewData3',
SIZE = 8192KB , FILEGROWTH = 10% )
TO FILEGROUP [NewFileGroup2]
GO

ALTER DATABASE [dp300mia]
ADD FILE (NAME = N'NewData4',
SIZE = 8192KB , FILEGROWTH = 10% )
TO FILEGROUP [NewFileGroup2]
GO

-- Alter objects' filegroups

EXEC sp_help 'dbo.myTable'

ALTER DATABASE [dp300mia]
MODIFY FILEGROUP [NewFileGroup2]
DEFAULT

CREATE TABLE myTable
(intmyTable2 INT PRIMARY KEY IDENTITY(1,1))

EXEC sp_help 'dbo.myTable'

CREATE TABLE myTable2
(intmyTable2 INT PRIMARY KEY IDENTITY(1,1))
ON [NewFileGroup]

EXEC sp_help 'dbo.myTable2'

CREATE NONCLUSTERED INDEX idx_myTable2 ON dbo.myTable2 (intmyTable2)
ON [NewFileGroup]

CREATE TABLE myTable3
(intmyTable2 INT PRIMARY KEY IDENTITY(1,1))

EXEC sp_help 'dbo.myTable3'
```

## 73. Manage schedules

```
USE msdb ;
GO

EXEC sp_add_schedule
    @schedule_name = N'ScheduleName' ,
    @freq_type = 4,
    @freq_interval = 1,  -- Fairly complex
    @active_start_time = 012345 ;
GO
```

```
EXEC sp_attach_schedule
   @job_name = N'JobName',
   @schedule_name = N'ScheduleName' ;
GO

--To view schedules:
USE msdb ;
GO
select *
from sysschedules
```

## 78. Create alerts for server configuration changes

```
sp_configure
GO
sp_configure 'show advanced options', 1
GO
RECONFIGURE
GO
sp_configure 'default trace enabled', 1
GO
RECONFIGURE
GO
sp_configure
GO
```

## 49. Identify sessions that cause blocking

```
-- Session 1
BEGIN TRANSACTION
UPDATE [dbo].[myTable]
SET [objectname] = 'sys.rscols'
where [objectname] = 'sysrscols'

rollback tran

-- Session 2
BEGIN TRANSACTION
UPDATE [dbo].[myTable]
SET [objectname] = 'sysrscols'
where [objectname] = 'sys.rscols'
```

## 54. Configure Resource Governor for performance

```
-- Enable Resource Governor
ALTER RESOURCE GOVERNOR RECONFIGURE;
GO
-- Create resource pools

CREATE RESOURCE POOL pDaytime
WITH (MAX_CPU_PERCENT = 20);
-- MIN_/MAX_CPU_PERCENT
-- CAP_CPU_PERECENT
-- MIN_/MAX_MEMORY_PERCENT
-- MIN_/MAX_IOPS_PER_VOLUME
GO
CREATE RESOURCE POOL pNighttime
WITH (MAX_CPU_PERCENT = 50);
GO
```

```
ALTER RESOURCE GOVERNOR RECONFIGURE;
GO

-- Create workload groups

CREATE WORKLOAD GROUP gDaytime
USING pDaytime;
GO
CREATE WORKLOAD GROUP gNighttime
USING pNighttime;
GO
ALTER RESOURCE GOVERNOR RECONFIGURE;
GO

-- Create Classifier Function

USE master
GO

CREATE FUNCTION ClassifierFunction()
RETURNS sysname
WITH SCHEMABINDING
AS
BEGIN
if DATEPART(HOUR,GETDATE())<8 or DATEPART(HOUR,GETDATE())>17
       BEGIN
              RETURN 'gNighttime';
       END
RETURN 'gDaytime';
END

-- Register this classified function:
ALTER RESOURCE GOVERNOR with (CLASSIFIER_FUNCTION = dbo.ClassifierFunction);
ALTER RESOURCE GOVERNOR RECONFIGURE;
GO

-- DMVs

SELECT * FROM sys.resource_governor_configuration

SELECT * FROM sys.dm_resource_governor_resource_pools

SELECT * FROM sys.dm_resource_governor_workload_groups
```

## 108, 110. Perform database and transaction log backups with options

```
BACKUP DATABASE NameOfDatabase
[FILEGROUP = 'X', FILEGROUP = 'Y' …]

TO MyPreviouslyCreatedNamedBackupDevice = 'BackupDevice'

MIRROR TO AnotherBackupDevice

WITH
       -- Backup Set Options
       COPY_ONLY
    DIFFERENTIAL
       COMPRESSION | NO_COMPRESSION
```

```
        DESCRIPTION = 'description'
        NAME = 'BackupSetName'
        CREDENTIAL
        ENCRYPTION
        FILE_SNAPSHOT [EXPIREDATE = 'Dec 31, 2029 11:59 PM' | RETAINDAYS = days]

        -- Media Set Options
        NOINIT | INIT
        NOSKIP | SKIP
        NOFORMAT | FORMAT

        -- Error Management Options
        NO_CHECKSUM | CHECKSUM
        STOP_ON_ERROR | CONTINUE_AFTER_ERROR

        -- Monitoring Options
        STATS = X

        -- Tape Options
        REWIND | NOREWIND
        UNLOAD | NOUNLOAD

;

BACKUP LOG NameOfDatabase
   TO MyPreviouslyCreatedNamedBackupDevice
WITH NORECOVERY, NO_TRUNCATE -- STANDBY = 'StandbyFileName'
```

## 109. Perform restore of user databases

```
RESTORE DATABASE NameOfDatabase
FROM MyPreviouslyCreatedNamedBackupDevice
-- WITH RECOVERY | NORECOVERY
-- FILE = BackupSetFileNumber

-- STOPAT = { 'datetime'| @datetime_var }
-- | STOPATMARK or STOPBEFOREMARK = { MarkName | LSNNumber } [ AFTER 'datetime']

-- Examples

RESTORE [dp300mi] FROM MyPreviouslyCreatedNamedBackupDevice
WITH FILE = 6, NORECOVERY, STOPAT = 'Jun 19, 2024 12:00 PM';
RESTORE [dp300mi] FROM MyPreviouslyCreatedNamedBackupDevice WITH FILE = 9, RECOVERY;

RESTORE VERIFYONLY FROM MyPreviouslyCreatedNamedBackupDevice

-- Restore from MI
RESTORE DATABASE NameOfDatabase
 FROM URL = 'https:// … ' , 'https:// … '
```

## Creating an Availability Group – additional permissions

```
USE [master]
GO
CREATE LOGIN [NT AUTHORITY\SYSTEM] FROM WINDOWS WITH DEFAULT_DATABASE=[master]
GO

GRANT ALTER ANY AVAILABILITY GROUP TO [NT AUTHORITY\SYSTEM]
GO
GRANT CONNECT SQL TO [NT AUTHORITY\SYSTEM]
```

```
GO
GRANT VIEW SERVER STATE TO [NT AUTHORITY\SYSTEM]
GO



GRANT CONNECT ON ENDPOINT::hadr_endpoint TO [NT Service\MSSQLSERVER]
GO
ALTER ENDPOINT hadr_endpoint STATE=STOPPED
ALTER ENDPOINT hadr_endpoint STATE=STARTED

select r.replica_server_name, r.endpoint_url,
rs.connected_state_desc, rs.last_connect_error_description,
rs.last_connect_error_number, rs.last_connect_error_timestamp
from sys.dm_hadr_availability_replica_states rs join sys.availability_replicas r
on rs.replica_id=r.replica_id



SELECT e.name AS mirror_endpoint_name
    ,s.name AS login_name
    ,p.permission_name
    ,p.state_desc AS permission_state
    ,e.state_desc endpoint_state
FROM sys.server_permissions p
INNER JOIN sys.endpoints e ON p.major_id = e.endpoint_id
INNER JOIN sys.server_principals s ON p.grantee_principal_id = s.principal_id
WHERE p.class_desc = 'ENDPOINT'
    AND e.type_desc = 'DATABASE_MIRRORING'

SELECT ar.replica_server_name,ag.name AS ag_name,ar.owner_sid,sp.name
          FROM sys.availability_replicas ar
          LEFT JOIN sys.server_principals sp
          ON sp.sid = ar.owner_sid
          INNER JOIN sys.availability_groups ag
          ON ag.group_id = ar.group_id
          WHERE ar.replica_server_name = SERVERPROPERTY('ServerName') ;
```