**1) To get the SQL Server version installed in your server**

SELECT @@VERSION AS SQLVersion

# **Latest updates for Microsoft SQL Server .**

<https://docs.microsoft.com/en-us/sql/database-engine/install-windows/latest-updates-for-microsoft-sql-server?view=sql-server-ver15>

**2) get statistics of all objects of an entire database in SQL Server**

select a.id as 'ObjectID', isnull(a.name,'Heap') as 'IndexName', b.name as 'TableName',

stats\_date (id,indid) as stats\_last\_updated\_time

from sys.sysindexes as a

inner join sys.objects as b

on a.id = b.object\_id where b.type = 'U'

**3) Check index fragmentation in SQL Server**

SELECT S.name as 'Schema',

T.name as 'Table',

I.name as 'Index',

DDIPS.avg\_fragmentation\_in\_percent,

DDIPS.page\_count

FROM sys.dm\_db\_index\_physical\_stats (DB\_ID(), NULL, NULL, NULL, NULL) AS DDIPS

INNER JOIN sys.tables T on T.object\_id = DDIPS.object\_id

INNER JOIN sys.schemas S on T.schema\_id = S.schema\_id

INNER JOIN sys.indexes I ON I.object\_id = DDIPS.object\_id

AND DDIPS.index\_id = I.index\_id

WHERE DDIPS.database\_id = DB\_ID()

and I.name is not null

AND DDIPS.avg\_fragmentation\_in\_percent > 0

ORDER BY DDIPS.avg\_fragmentation\_in\_percent desc

**4) Check unused indexes**

**Performance Tuning is quite interesting and Index plays a vital role in it. A proper index can improve the performance and a bad index can hamper the performance. Here is the script from my script bank, which I use to identify unused indexes on any database.**

SELECT TOP 25

o.name AS ObjectName

, i.name AS IndexName

, i.index\_id AS IndexID

, dm\_ius.user\_seeks AS UserSeek

, dm\_ius.user\_scans AS UserScans

, dm\_ius.user\_lookups AS UserLookups

, dm\_ius.user\_updates AS UserUpdates

, p.TableRows

, 'DROP INDEX ' + QUOTENAME(i.name)

+ ' ON ' + QUOTENAME(s.name) + '.'

+ QUOTENAME(OBJECT\_NAME(dm\_ius.OBJECT\_ID)) AS 'drop statement'

FROM sys.dm\_db\_index\_usage\_stats dm\_ius

INNER JOIN sys.indexes i ON i.index\_id = dm\_ius.index\_id

AND dm\_ius.OBJECT\_ID = i.OBJECT\_ID

INNER JOIN sys.objects o ON dm\_ius.OBJECT\_ID = o.OBJECT\_ID

INNER JOIN sys.schemas s ON o.schema\_id = s.schema\_id

INNER JOIN (SELECT SUM(p.rows) TableRows, p.index\_id, p.OBJECT\_ID

FROM sys.partitions p GROUP BY p.index\_id, p.OBJECT\_ID) p

ON p.index\_id = dm\_ius.index\_id AND dm\_ius.OBJECT\_ID = p.OBJECT\_ID

WHERE OBJECTPROPERTY(dm\_ius.OBJECT\_ID,'IsUserTable') = 1

AND dm\_ius.database\_id = DB\_ID()

AND i.type\_desc = 'nonclustered'

AND i.is\_primary\_key = 0

AND i.is\_unique\_constraint = 0

ORDER BY (dm\_ius.user\_seeks + dm\_ius.user\_scans + dm\_ius.user\_lookups) ASC

**5) We can learn that all databases CPU resources usage.**

WITH DB\_CPU\_STATS\_ON\_INSTANCE

AS

(SELECT DatabaseID, DB\_Name(DatabaseID) AS [DatabaseName], SUM(total\_worker\_time) AS [CPU\_Time\_Ms]

FROM sys.dm\_exec\_query\_stats AS qs

CROSS APPLY (SELECT CONVERT(int, value) AS [DatabaseID]

FROM sys.dm\_exec\_plan\_attributes(qs.plan\_handle)

WHERE attribute = N'dbid') AS F\_DB

GROUP BY DatabaseID)

SELECT ROW\_NUMBER() OVER(ORDER BY [CPU\_Time\_Ms] DESC) AS [row\_num],

DatabaseName, [CPU\_Time\_Ms],

CAST([CPU\_Time\_Ms] \* 1.0 / SUM([CPU\_Time\_Ms]) OVER() \* 100.0 AS DECIMAL(5, 2)) AS [CPUPercent]

FROM DB\_CPU\_STATS\_ON\_INSTANCE

WHERE DatabaseID > 4

AND DatabaseID <> 32767

ORDER BY row\_num OPTION (RECOMPILE);

**6) The MSDB Suspect\_pages table records suspect pages. Regular monitoring of the MSDB Suspect\_pages table is advised ,  particularly for 823 errors ,824 errors,Bad CheckSum and Torn Page.**

SELECT DB\_NAME(database\_id),[file\_id],page\_id,

CASE event\_type

WHEN 1 THEN '823 or 824 or Torn Page'

WHEN 2 THEN 'Bad Checksum'

WHEN 3 THEN 'Torn Page'

WHEN 4 THEN 'Restored'

WHEN 5 THEN 'Repaired (DBCC)'

WHEN 7 THEN 'Deallocated (DBCC)'

END,

error\_count,

last\_update\_date

FROM msdb..suspect\_pages

# **7) SQL Server Cached blocking history with sys.dm\_db\_index\_operational\_stats**

**Transactions generate IO, latching  and  locking on tables and indexes  , while attempting to access data. The**sys.dm\_db\_index\_operational\_stats**DMV returns aggregated data on this activity.**

select db\_name(database\_id) DB,

QUOTENAME(OBJECT\_SCHEMA\_NAME(object\_id, database\_id))

+ N'.'

+ QUOTENAME(OBJECT\_NAME(object\_id, database\_id)) ObjDetails,

row\_lock\_wait\_in\_ms + page\_lock\_wait\_in\_ms Block\_Wait\_Time\_in\_ms

from sys.dm\_db\_index\_operational\_stats(NULL,NULL,NULL,NULL)

order by Block\_Wait\_Time\_in\_ms desc,ObjDetails desc