Calculating space used by table and per column

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:27 AM PST

Contents

Scripts to calculate the space used per table, index, and column

This is a "How To" article related to space and storage management.

Calculate the size of each table (rowcount and storage)

```
SELECT
    o.name AS table_name,
    s.name AS schema_name,
    p.rows AS row_count,
    SUM(a.total_pages) * 8 AS allocated_pages_KB,
    SUM(a.used_pages) * 8 AS used_pages_KB,
    SUM(a.data_pages) * 8 AS data_pages_KB,
    (SUM(a.total_pages) - SUM(a.used_pages)) * 8 AS unused_pages_KB
FROM
    sys.all_objects o
    INNER JOIN sys.indexes i ON o.OBJECT_ID = i.object_id
    INNER JOIN sys.partitions p ON i.object_id = p.OBJECT_ID AND i.index_id = p.index_id
    INNER JOIN sys.allocation_units a ON p.partition_id = a.container_id
    LEFT OUTER JOIN sys.schemas s ON o.schema_id = s.schema_id
WHERE
    (o.type = 'U' or o.type like 'V%') and o.is_ms_shipped <> 1
GROUP BY
    o.Name, s.Name, p.Rows
ORDER BY
    o.Name;
```

Sample output:

Results Messages

	table_name	schema_name	row_count	allocated_pages_KB	used_pages_KB	data_pages_KB	unused_pages_KB
1	Address	Person	19614	5600	5272	5184	328
2	AddressType	Person	6	216	48	24	168
3	AWBuildVersion	dbo	1	72	16	8	56
4	BillOfMaterials	Production	2679	792	360	312	432
5	BusinessEntity	Person	20777	1488	1336	1304	152
6	BusinessEntityAddress	Person	19614	2656	2312	2248	344
7	BusinessEntityContact	Person	909	800	192	128	608
8	ContactType	Person	20	144	32	16	112

Calculate the size of each table and index (rowcount and storage)

The important details can be retrieved from either `sys.dm_db_partition_stats` or `sys.allocation_units`:

```
SELECT
    o.name AS table_name,
    s.name AS schema_name,
    i.index_id,
    i.type_desc AS IndexType,
    i.name AS indexname,
    ps.row_count,
    ps.reserved_page_count * 8 AS allocated_pages_KB,
    ps.used_page_count * 8 AS used_pages_KB,
    ps.in_row_data_page_count * 8 AS data_pages_KB,
    (ps.reserved_page_count - ps.used_page_count) * 8 AS unused_pages_KB,
    ps.in_row_reserved_page_count * 8 AS in_row_allocated_pages_KB,
    ps.lob_reserved_page_count * 8 AS lob_allocated_pages_KB,
    ps.row_overflow_reserved_page_count * 8 AS row_overflow_allocated_pages_KB,
    ps.in_row_used_page_count * 8 AS in_row_used_pages_KB,
    ps.in_row_data_page_count * 8 AS in_row_data_pages_KB,
    ps.lob_used_page_count * 8 AS lob_used_pages_KB,
    ps.row_overflow_used_page_count * 8 AS row_overflow_used_pages_KB
FROM
    sys.all_objects o
    INNER JOIN sys.indexes i ON o.object_id = i.object_id
    INNER JOIN sys.dm_db_partition_stats ps ON ps.[object_id] = i.[object_id] AND ps.index_id = i.index_id
    LEFT OUTER JOIN sys.schemas s ON o.schema_id = s.schema_id
WHERE
    (o.type = 'U' or o.type like 'V%') and o.is_ms_shipped <> 1
ORDER BY
    o.name, i.index_id;
```

Sample output:

Results | Messages

	table_name	schema_...	index_id	IndexType	indexname	row_count	allocated_pages_KB	used_pages_KB	data_pages_KB	unused_pages_KB
1	Address	Person	1	CLUSTERED	PK_Address_AddressID	19614	2824	2728	2712	96
2	Address	Person	2	NONCLUSTERED	AK_Address_rowguid	19614	584	528	512	56
3	Address	Person	3	NONCLUSTERED	IX_Address_AddressLine1_Addr...	19614	1800	1728	1688	72
4	Address	Person	4	NONCLUSTERED	IX_Address_StateProvinceID	19614	392	288	272	104
5	AddressType	Person	1	CLUSTERED	PK_AddressType_AddressTypeID	6	72	16	8	56
6	AddressType	Person	2	NONCLUSTERED	AK_AddressType_Name	6	72	16	8	56
7	AddressType	Person	3	NONCLUSTERED	AK_AddressType_rowguid	6	72	16	8	56
8	AWBuildVersion	dbo	1	CLUSTERED	PK_AWBuildVersion_SystemInfo...	1	72	16	8	56
9	BillOfMaterials	Production	1	CLUSTERED	AK_BillOfMaterials_ProductAsse...	2679	328	176	160	152
10	BillOfMaterials	Production	2	NONCLUSTERED	PK_BillOfMaterials_BillOfMaterial...	2679	200	88	72	112
11	BillOfMaterials	Production	3	NONCLUSTERED	IX_BillOfMaterials_UnitMeasureC...	2679	264	96	80	168

in_row_allocated_pages_KB	lob_allocated_pages_KB	row_overflow_allocated_pages_KB	in_row_used_pages_KB	in_row_data_pages_KB	lob_used_pages_KB	row_overflow_used_pages_KB
2824	0	0	2728	2712	0	0
584	0	0	528	512	0	0
1800	0	0	1728	1688	0	0
392	0	0	288	272	0	0
72	0	0	16	8	0	0
72	0	0	16	8	0	0
72	0	0	16	8	0	0
72	0	0	16	8	0	0
328	0	0	176	160	0	0
200	0	0	88	72	0	0
264	0	0	96	80	0	0

```
-- similar output with fewer details, retrieved from a different source to cross-check:
SELECT
    o.name AS table_name,
    s.Name AS schema_name,
    i.index_id,
    i.type_desc AS index_type,
    i.name AS index_name,
    p.rows AS rowcounts,
    SUM(a.total_pages) * 8 AS allocated_pages_KB,
    SUM(a.used_pages) * 8 AS used_pages_KB,
    SUM(a.data_pages) * 8 AS data_pages_KB,
    (SUM(a.total_pages) - SUM(a.used_pages)) * 8 AS unused_pages_KB
FROM
    sys.all_objects o
    INNER JOIN sys.indexes i ON o.OBJECT_ID = i.object_id
    INNER JOIN sys.partitions p ON i.object_id = p.OBJECT_ID AND i.index_id = p.index_id
    INNER JOIN sys.allocation_units a ON p.partition_id = a.container_id
    LEFT OUTER JOIN sys.schemas s ON o.schema_id = s.schema_id
WHERE
    (o.type = 'U' or o.type like 'V%') and o.is_ms_shipped <> 1
GROUP BY
    o.Name, s.Name, i.type_desc, i.name, i.index_id, p.Rows
ORDER BY o.Name, i.index_id;
```

Sample output:

Results Messages

	table_name	schema_n...	index_id	index_type	index_name	rowcounts	allocated_pages_KB	used_pages_KB	data_pages_KB	unused_pages_KB
1	Address	Person	1	CLUSTERED	PK_Address_AddressID	19614	2824	2728	2712	96
2	Address	Person	2	NONCLUSTERED	AK_Address_rowguid	19614	584	528	512	56
3	Address	Person	3	NONCLUSTERED	IX_Address_AddressLine1_AddressLin...	19614	1800	1728	1688	72
4	Address	Person	4	NONCLUSTERED	IX_Address_StateProvinceID	19614	392	288	272	104
5	AddressType	Person	1	CLUSTERED	PK_AddressType_AddressTypeID	6	72	16	8	56
6	AddressType	Person	2	NONCLUSTERED	AK_AddressType_Name	6	72	16	8	56
7	AddressType	Person	3	NONCLUSTERED	AK_AddressType_rowguid	6	72	16	8	56
8	AWBuildVersion	dbo	1	CLUSTERED	PK_AWBuildVersion_SystemInformatio...	1	72	16	8	56
9	BillOfMaterials	Production	1	CLUSTERED	AK_BillOfMaterials_ProductAssemblyID...	2679	328	176	160	152
10	BillOfMaterials	Production	2	NONCLUSTERED	PK_BillOfMaterials_BillOfMaterialsID	2679	200	88	72	112
11	BillOfMaterials	Production	3	NONCLUSTERED	IX_BillOfMaterials_UnitMeasureCode	2679	264	96	80	168

Calculate the size of each table and index (with index usage statistics where available)

```
SELECT
    o.name AS table_name,
    s.name AS schema_name,
    i.index_id,
    i.type_desc AS IndexType,
    i.name AS indexname,
    ps.row_count,
    ps.reserved_page_count * 8 AS allocated_pages_KB,
    ps.used_page_count * 8 AS used_pages_KB,
    ps.in_row_data_page_count * 8 AS data_pages_KB,
    (ps.reserved_page_count - ps.used_page_count) * 8 AS unused_pages_KB,
    ius.user_seeks, ius.user_scans, ius.user_lookups, ius.user_updates,
    ius.last_user_seek, ius.last_user_scan, ius.last_user_lookup, ius.last_user_update
FROM
    sys.all_objects o
    INNER JOIN sys.indexes i ON o.object_id = i.object_id
    INNER JOIN sys.dm_db_partition_stats ps ON ps.[object_id] = i.object_id AND ps.index_id = i.index_id
    LEFT JOIN sys.dm_db_index_usage_stats ius ON ius.object_id = i.object_id AND ius.index_id = i.index_id
    LEFT JOIN sys.schemas s ON o.schema_id = s.schema_id
WHERE
    (o.type = 'U' or o.type like 'V%') and o.is_ms_shipped <> 1
ORDER BY
    o.name, i.index_id;
```

Sample output:

Results Messages

	table_name	schema_...	index_id	IndexType	indexname	row_count	allocated_pages_KB	used_pages_KB	data_pages_KB	unused_pages_KB
1	Address	Person	1	CLUSTERED	PK_Address_AddressID	19614	2824	2728	2712	96
2	Address	Person	2	NONCLUSTERED	AK_Address_rowguid	19614	584	528	512	56
3	Address	Person	3	NONCLUSTERED	IX_Address_AddressLine1_Address...	19614	1800	1728	1688	72
4	Address	Person	4	NONCLUSTERED	IX_Address_StateProvinceID	19614	392	288	272	104
5	AddressType	Person	1	CLUSTERED	PK_AddressType_AddressTypeID	6	72	16	8	56
6	AddressType	Person	2	NONCLUSTERED	AK_AddressType_Name	6	72	16	8	56
7	AddressType	Person	3	NONCLUSTERED	AK_AddressType_rowguid	6	72	16	8	56
8	AWBuildVersion	dbo	1	CLUSTERED	PK_AWBuildVersion_SystemInform...	1	72	16	8	56
9	BillOfMaterials	Production	1	CLUSTERED	AK_BillOfMaterials_ProductAssembl...	2679	328	176	160	152
10	BillOfMaterials	Production	2	NONCLUSTERED	PK_BillOfMaterials_BillOfMaterialsID	2679	200	88	72	112
11	BillOfMaterials	Production	3	NONCLUSTERED	IX_BillOfMaterials_UnitMeasureCode	2679	264	96	80	168

user_seeks	user_scans	user_lookups	user_updates	last_user_seek	last_user_scan	last_user_lookup	last_user_update
0	5	1	0	NULL	2022-10-21 07:38:20.843	2022-10-21 07:38:37.970	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	0	0	0	2022-10-21 07:38:37.970	NULL	NULL	NULL
0	2	1	0	NULL	2022-10-21 07:39:38.970	2022-10-21 07:40:10.270	NULL
1	0	0	0	2022-10-21 07:40:10.270	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
0	1	1	0	NULL	2022-10-21 07:40:43.250	2022-10-21 07:41:16.640	NULL
1	0	0	0	2022-10-21 07:41:16.640	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Calculate the size of the column data stored in a table

This query runs on a per-table basis, because the calculation is very expensive. Set the requested table name as "schema.table" e.g. "dbo.table1" into the @ObjectTable variable.

```
DECLARE @ObjectTable     sysname = 'Person.Person'
DECLARE @Name            sysname
DECLARE @Type            sysname
DECLARE @Length          INT
DECLARE @Total           BIGINT = 0
DECLARE @SQL             NVARCHAR(4000)
DECLARE @ParamDefinition NVARCHAR(100)= N'@Total BIGINT OUTPUT'

SET ANSI_WARNINGS OFF

DECLARE vColumns CURSOR FOR
    SELECT c.name, t.name as 'type', c.max_length
    FROM sys.columns c
    INNER JOIN sys.types t ON c.system_type_id = t.user_type_id
    WHERE object_id = OBJECT_ID(@ObjectTable)
    ORDER BY c.column_id

OPEN vColumns
FETCH NEXT FROM vColumns INTO @Name, @Type, @Length

WHILE @@FETCH_STATUS = 0
BEGIN
    SET @Total = 0
    SET @SQL = 'SELECT @Total = SUM(DATALENGTH(' + @Name + ')) FROM ' + @ObjectTable
    EXECUTE sp_executesql @SQL, @ParamDefinition, @Total = @Total OUTPUT;
    PRINT + 'Size (KB): ' + CONVERT(CHAR(20), (@Total / 1024)) + 'Column: '+ CONVERT(VARCHAR(128), @Name) + '
    FETCH NEXT FROM vColumns INTO @Name, @Type, @Length
END

CLOSE vColumns;
DEALLOCATE vColumns;

/* sample output:
Size (KB): 78                Column: BusinessEntityID (type: int, length: 4)
Size (KB): 78                Column: PersonType (type: nchar, length: 4)
Size (KB): 19                Column: NameStyle (type: bit, length: 1)
Size (KB): 5                 Column: Title (type: nvarchar, length: 16)
Size (KB): 230               Column: FirstName (type: nvarchar, length: 100)
Size (KB): 23                Column: MiddleName (type: nvarchar, length: 100)
Size (KB): 218               Column: LastName (type: nvarchar, length: 100)
Size (KB): 0                 Column: Suffix (type: nvarchar, length: 20)
Size (KB): 78                Column: EmailPromotion (type: int, length: 4)
Size (KB): 19                Column: AdditionalContactInfo (type: xml, length: -1)
Size (KB): 24073             Column: Demographics (type: xml, length: -1)
Size (KB): 312               Column: rowguid (type: uniqueidentifier, length: 16)
Size (KB): 156               Column: ModifiedDate (type: datetime, length: 8)
*/
```

Public Doc Reference

- Blog post: Lesson Learned 150: Calculating the space used by table and per column

How good have you found this content?