Analysing Indexes Part 2

Here's the next instalment, expect there to at least another two or three.

- I'm still only working with basic indexing
- No partitioning
- I'm not looking at xml indexes.

14.0 Collecting Raw Index usage data

- Collects table name, index name , reads and writes where there is a value for reads or writes
- Collect daily / hourly or as required
    - The greater the granularity the greater volume of data collected
- Values stored in the dmv's are cumulative so we have to store snapshots and query for weekly/daily/hourly differences to extract usage figures.

14.1 Table to store data – create in a monitoring database.

- It's best to create a dedicated database to collect data, most DBA's have their own database
- For the examples here we'll assume it's called ServerAdmin

```
--
-- create in ServerAdmin database
--
Use ServerAdmin;
go
IF  EXISTS (SELECT * FROM sys.objects WHERE
objectproperty(OBJECT_ID(N'[dbo].[tbl_IndexActivity]'),'IsUserTable') = 1)
DROP TABLE [dbo].[tbl_IndexActivity];
--
create table dbo.tbl_IndexActivity
(
TableName sysname not null,
IndexName sysname not null,
IndexType varchar(4) not null,
Reads bigint not null,
Writes bigint not null,
TheDate datetime  constraint DF_IndexActivity_TheDate default getdate() not null
);
```

14.2 Run this query in a scheduled job

- I don't actually create a stored procedure here as I would have to create it within the monitored database, something I might not want to do.

```
--
-- Run this is the context of the database to be analysed
-- every night for daily stats
--
insert into
ServerAdmin.dbo.tbl_IndexActivity(TableName,IndexName,IndexType,Reads,Writes)
```

```
select object_name(s.object_id) as TableName, isnull(i.name,'HEAP') as IndexName,
case i.index_id
when 0 then 'HEAP'
when 1 then 'CLUS'
else 'NC'
end as IndexType
,reads=user_seeks + user_scans + user_lookups
,writes =  user_updates
from sys.dm_db_index_usage_stats s join sys.indexes i
on s.object_id = i.object_id and i.index_id = s.index_id
where objectproperty(s.object_id,'IsUserTable') = 1
and s.database_id = db_id()
order by reads desc;
go
```

- This will create lots of data as it will create a row for each index which has been accessed
- Remember that dmv's contain cumulative data so we may find some indexes have not been used recently.
- Once we have sets of data we can view the difference to see what's happening
- Although I've suggested this collection to run daily the interval could be any suitable value down to a couple of minutes for real time stats.
- I'm using this basic data collection to decide which indexes I want to analyse in greater detail.

14.3 Most Writes for an hour period

- Example assumes collection of data hourly
- Modify the SARGs for ia1.thedate and ia2.thedate to return different data sets.
- You may wish to to collect top 25 results in each category and store in another table
- This data would make a good OLAP cube

```
--
-- Example query to extract index usage
-- Results for the hour 10:00 to 11:00 today
-- Highest Writes
--
select  ia1.tablename,ia1.indexname,ia2.reads-ia1.reads as Reads,ia2.writes-
ia1.writes as Writes
from ServerAdmin.dbo.tbl_indexactivity ia1 join ServerAdmin.dbo.tbl_indexactivity
ia2 on
ia1.tablename = ia2.tablename and ia1.indexname=ia2.indexname
where datepart(hh,ia1.thedate) = 10  and convert(char(8),ia1.thedate,112) =
convert(char(8),getdate(),112)
and  datepart(hh,ia2.thedate) = 11 and convert(char(8),ia2.thedate,112) =
convert(char(8),getdate(),112)
and ( (ia2.reads-ia1.reads) + (ia2.writes-ia1.writes) ) >0
order by (ia2.writes-ia1.writes) desc;
go
```

Abridged results

- Table names and index names are fictitious

tablename	indexname	Reads	Writes
TransportLinks	PK_TransportLinks	153575	188427
EuropeSites	Idx_EuropeSites_Manager	42112	57345
RegionalManager	Uk_RegionalManager_Name	49716	50318
LocalAreaSales	Idx_LocalAreaSales_One	0	41580
LocalAreaSales	Idx_LocalAreaSales_Two	0	41580

14.4 Most Reads for an hour period

- Assumes collection of data hourly

```
 --
-- Example query to extract index usage
-- Results for the hour 10:00 to 11:00 today
-- Highest Reads
--
select  ia1.tablename,ia1.indexname,ia2.reads-ia1.reads as Reads,ia2.writes-
ia1.writes as Writes
from ServerAdmin.dbo.tbl_indexactivity ia1 join ServerAdmin.dbo.tbl_indexactivity
ia2 on
ia1.tablename = ia2.tablename and ia1.indexname=ia2.indexname
where datepart(hh,ia1.thedate) = 10  and convert(char(8),ia1.thedate,112) =
convert(char(8),getdate(),112)
and  datepart(hh,ia2.thedate) = 11 and convert(char(8),ia2.thedate,112) =
convert(char(8),getdate(),112)
and ( (ia2.reads-ia1.reads) + (ia2.writes-ia1.writes) ) >0
order by (ia2.reads-ia1.reads) desc;
go
```

Abridged results

tablename	indexname	Reads	Writes
SalesItems	Uk_SalesItems_StockNumber	264492	4
SalesItems	Idx_SalesItems_Reject	257247	0
EmployeeGroup	PK_EmployeeGroup	212268	11
TransportLinks	PK_TransportLinks	153575	188427
RestrictedItems	Idx_RestrictedItems_Analysis	76818	38608

14.5 Most Reads and Writes for an hour period

```
 --
-- Example query to extract index usage
-- Results for the hour 10:00 to 11:00 today
-- Highest Reads and Writes
--
select  ia1.tablename,ia1.indexname,ia2.reads-ia1.reads as Reads,ia2.writes-
ia1.writes as Writes
from ServerAdmin.dbo.tbl_indexactivity ia1 join ServerAdmin.dbo.tbl_indexactivity
ia2 on
ia1.tablename = ia2.tablename and ia1.indexname=ia2.indexname
where datepart(hh,ia1.thedate) = 10  and convert(char(8),ia1.thedate,112) =
convert(char(8),getdate(),112)
and  datepart(hh,ia2.thedate) = 11 and convert(char(8),ia2.thedate,112) =
convert(char(8),getdate(),112)
and ( (ia2.reads-ia1.reads) + (ia2.writes-ia1.writes) ) >0
order by ((ia2.reads-ia1.reads)+(ia2.writes-ia1.writes)) desc;
```

```
go
```

Abridged results

tablename	indexname	Reads	Writes
TransportLinks	PK_TransportLinks	153575	188427
SalesItems	Uk_SalesItems_StockNumber	264492	4
SalesItems	Idx_SalesItems_Reject	257247	0
EmployeeGroup	PK_EmployeeGroup	212268	11
RestrictedItems	Idx_RestrictedItems_Analysis	76818	38608
NearestPub	Uk_NearestPub	49716	50318

14.6 Most Reads and Writes showing %age proportion for an hour period

```
 --
-- Example query to extract index usage
-- Results for the hour 10:00 to 11:00 today
-- Highest Reads and Writes with proportion shown
--
select  ia1.tablename,ia1.indexname,ia2.reads-ia1.reads as Reads,ia2.writes-
ia1.writes as Writes
, convert(int, (ia2.reads-ia1.reads)*1.0/((ia2.reads-ia1.reads)+(ia2.writes-
ia1.writes))*100.0) as 'Read%'
, convert(int, (ia2.writes-ia1.writes)*1.0/((ia2.reads-ia1.reads)+(ia2.writes-
ia1.writes))*100.0) as 'Write%'
from ServerAdmin.dbo.tbl_indexactivity ia1 join ServerAdmin.dbo.tbl_indexactivity
ia2 on
ia1.tablename = ia2.tablename and ia1.indexname=ia2.indexname
where datepart(hh,ia1.thedate) = 10  and convert(char(8),ia1.thedate,112) =
convert(char(8),getdate(),112)
and  datepart(hh,ia2.thedate) = 11 and convert(char(8),ia2.thedate,112) =
convert(char(8),getdate(),112)
and ( (ia2.reads-ia1.reads) + (ia2.writes-ia1.writes) ) >100
order by ((ia2.reads-ia1.reads)+(ia2.writes-ia1.writes)) desc;
go
```

Abridged results

tablename	indexname	Reads	Writes	Read %	Write %
TransportLinks	PK_TransportLinks	153575	188427	44	55
SalesItems	Uk_SalesItems_StockNumber	264492	4	99	0
SalesItems	Idx_SalesItems_Reject	257247	0	100	0
EmployeeGroup	PK_EmployeeGroup	212268	11	99	0
RestrictedItems	Idx_RestrictedItems_Analysis	76818	38608	66	33
NearestPub	Uk_NearestPub	49716	50318	49	50

- So this allows us to view index usage at the granularity of data collection.
- For the application I'm supporting data collection grows at 1,700 rows per hour, that's about 1.3 million rows per month
- This data will allow us to collect more detailed information from the dmv's to fully analyse index usage
- I'd probably suggest selecting the top 100 indexes for each category for a 7 day period and using this to extract more detailed statistics. (This is in Part 3)

15.0 So Just how many indexes do I have ?

```
--
-- How many indexes ?
-- run in context of database to where they are to be counted
--
select type_desc,count(*) from sys.indexes
where objectproperty(OBJECT_ID,'IsUserTable') = 1
group by type_desc;
go
```

type_desc	(No column name)
CLUSTERED	2936
HEAP	17
NONCLUSTERED	4067

15.1 How many empty tables do I have ?

- Many third party applications have unused sections which may result in many empty tables
- Having a definitive list of unused tables will help provide a filter to use in subsequent analysis
- As far as I can see this method is accurate, however, row count is not so easily found in SQL 2005, it's available within system functions which cannot be accessed normally.

15.2 Create this table in the ServerAdmin database

```
 USE [ServerAdmin];
GO
--
IF  EXISTS (SELECT * FROM sys.objects WHERE
objectproperty(OBJECT_ID(N'[dbo].[tbl_EmptyTables]'),'IsUserTable') = 1)
DROP TABLE [dbo].[tbl_EmptyTables];
--
create table dbo.tbl_EmptyTables
(
TableName sysname not null,
[Object_id] bigint not null,
IndexName sysname not null,
Row_Count bigint not null
);
go
```

15.3 Run this script in the context of the database to be analysed

- Extracts by clustered index and by HEAP
- Easiest to use dbo.sysindexes although not ideal.

```
 --
-- run in database to be analysed
--
insert into
ServerAdmin.dbo.tbl_EmptyTables(TableName,[Object_id],IndexName,Row_Count)
select object_name(id),id,name,rows from dbo.sysindexes
where rows = 0 and indid = 1
and objectproperty(ID,'IsUserTable') = 1;
--
insert into
ServerAdmin.dbo.tbl_EmptyTables(TableName,[Object_id],IndexName,Row_Count)
select object_name(id),id,isnull(name,'HEAP'),rows from dbo.sysindexes
where rows = 0 and indid = 0
and objectproperty(ID,'IsUserTable') = 1;
go
```

15.4 Add this unique index to check there are no duplicate entries

```
 use ServerAdmin;
go
--
create unique index uk_tbl_EmptyTables_Object_ID on
dbo.tbl_EmptyTables([object_id]);
go
```

- This produces a result set of over 2,000 for the application I am supporting

16.0 How many statistics?

- Now the viewing of statistics is somewhat different in SQL 2005
- We don't want to see stats on indexes
- You can also keep a watch on how many "auto stats" your database has with this query
- I generally consider high numbers of auto stats being indicative of inadequate indexing.
- I generally will periodically delete all system stats from a database
- If there were stats and you create an index the stats remain
- The stats may have come from who knows when and may not be relevant any more
- As always be careful, your database may take a temporary performance hit if it's running on auto stats
- The subject of statistics will be discussed later in this series

16.1 List all statistics

```
--
-- show statistics in the database
```

```
--
select object_name(s.object_id),* from sys.stats s
where objectproperty(s.OBJECT_ID,'IsUserTable') = 1
and s.auto_created | s.user_created = 1;
go
```

✓ The symbol in the AND statement (|) is SHIFT + Backslash (left of Z on my keyboard)

Abridged results

(No column name)	Object id	name	Stats id	Auto created	User created	No recompute
Employees	119887	sts_Employees1	2	0	1	0
Employees	119887	sts_Employees2	3	0	1	0
Sales	1187550	_WA_Sys_00000002_08B54D69	2	1	0	0
RegionSouth	1656045	sts_RegionSouth_Pole	3	0	1	0
RegionNorth	1656045	_WA_Sys_Rows_75D7831F	4	1	0	0

- System generated statistics always start **_WA_Sys_**
- There are also hypothetical indexes which you should never see, these start **hind_%**

16.2 How many statistics do I have ?

```
--
-- How many statistics ?
-- run in contect of database to whose stats are to be counted
--
select
case convert(char(1),auto_created)+convert(char(1),user_created)
when '10' then 'Auto Created'
when '01' then 'User Created'
else 'Both'
end as StatsType,
count(*) as HowMany from sys.stats s
where objectproperty(s.OBJECT_ID,'IsUserTable') = 1
and s.auto_created | s.user_created = 1
group by convert(char(1),auto_created)+convert(char(1),user_created);
go
```

StatsType	HowMany
User Created	7786
Auto Created	121

15.3 View to display Statistics Information

- It used to be quite simple to view statistics, as against indexes in SQL 2000

- Not so easy in SQL 2005
- This view which should be created in the user database that you wish to display basic information on statistics

```
--
-- view to provide information on statistics only
-- filters out index stats and system objects
-- per database create in database
--
IF  EXISTS (SELECT * FROM sys.views WHERE object_id =
OBJECT_ID(N'[dbo].[dm_db_statistics]'))
DROP VIEW [dbo].[dm_db_statistics];
go
create view dbo.dm_db_statistics
as
select object_name(s.[object_id]) as TableName,c.name as ColumnName,s.name as
StatName,s.auto_created,s.user_created,s.no_recompute,s.[object_id],
s.stats_id,sc.stats_column_id,sc.column_id,stats_date(s.[object_id], s.stats_id) as
LastUpdated
from sys.stats s join sys.stats_columns sc on sc.[object_id] = s.[object_id] and sc.stats_id =
s.stats_id
join sys.columns c on c.[object_id] = sc.[object_id] and c.column_id = sc.column_id
where objectproperty(s.OBJECT_ID,'IsUserTable') = 1
and s.auto_created | s.user_created = 1;
go
```

Sample results (please excuse formatting)

Table Name	Column Name	Stat Name	Auto created	User created	No recompute	Object id	Stats id	Stats column id	Column id	Last Updated
IndexStats	LastUpdate	_WA_Sys_ IndexStats _164452B1	1	0	0	373576369	2	1	10	2007-07-16 16:34:21.780
WeeklyStats	counter_name	_WA_Sys_counter_name_173876EA	1	0	0	389576426	2	1	3	2007-07-16 16:34:21.827
Product2007	header	_WA_Sys_header_24927208	1	0	0	613577224	2	1	1	2007-07-16 16:34:21.890
Trace20070707	TextData	_WA_Sys_00000003_2D27B809	1	0	0	757577737	2	1	3	2007-07-10 14:15:07.863

- When there are many indexes and statistics the results set can be large and make analysis difficult
- The application I support has over 15,000 indexes and statistics, many of these relate to unpopulated tables.
- Should the system procedure sp_createstats have been run against the database then statistics will have been created on empty tables.

- This can be a useful procedure in as much as you can command it to create statistics on columns of multi column or compound indexes
- However, great as this may sound, if you have a large number of covered indexes or precise compound indexes you might not actually benefit from these extra statistics – and these will have to be maintained – so use this proc with care.

http://msdn2.microsoft.com/en-us/library/ms186834.aspx