What is Index Fragmentation

- Unavoidable
- OLTP environment
- Causes more than the optimal amount of disk I/O to be performed in accessing a table
- Disk I/Os take longer than they optimally would

Most Common Causes of Index Fragmentation

- INSERT and UPDATE operations causing Page splits
- DELETE operations
- Large row size

Types of Index Fragmentation

- Internal Fragmentation
  - When pages are less than fully used, the part of each page that is unused constitutes a form of fragmentation
- Logical Fragmentation
  - Logical fragmentation occurs when the pages in the doubly linked list are not contiguous in the index, meaning that indexes have pages in which the logical ordering of pages, which is based on the key value, does not match the physical ordering inside the data file.
- Extent Fragmentation
  - Extent fragmentation occurs when the extents of a table or index are not contiguous with the database leaving extents from one or more indexes intermingled in the file.

How to Detect Index Fragmentation

- Use the sys.dm_db_index_physical_stats DMF, that replaces the deprecated DBCC SHOWCONTIG command
- This DMF has three modes
  - DETAILED - reads all data and index pages. Be careful with using this option since it causes the entire index to be read into memory and may result in IO/Memory issues.
  - SAMPLED: reads 1% of the pages if more than 10,000 pages.
  - LIMITED: only reads the parent level of b-tree (same as DBCC SHOWCONTIG WITH FAST). Limited option doesn't report page density, since it does not read the leaf level pages.
- Pay attention to the following columns in the output of this DMF
  - Avg_fragmentation_in_percent – depicts logical fragmentation
  - Avg_page_space_used_in_percent – depicts internal fragmentation

Solutions to address Index Fragmentation

- Recreate - CREATE INDEX WITH DROP_EXISTING
- Rebuild - ALTER INDEX ... REBUILD (replaces DBCC REINDEX)
- Reorganize - ALTER INDEX ... REORGANIZE (replaces DBCC INDEXDEFRAG)
- To reduce the extent fragmentation of a heap, create a clustered index on the table and then drop the index

Reorganize vs. Rebuilding indexes

#	Characteristic	Alter Index REORGANIZE	Alter Index REBUILD
1	Online or Offline	Online	Offline (unless using the Online keyword)
2	Address Internal Fragmentation	Yes (can only raise page density)	Yes
3	Address Logical Fragmentation	Yes	Yes
4	Transaction Atomicity	Small Discrete Transactions	Single Atomic Transaction
5	Rebuild Statistics Automatically	No	Yes
6	Parallel Execution in multi-processor machines	No	Yes
7	Untangle Indexes that have become interleaved within a data file	No	Yes
8	Transaction log space used	Less	More
9	Additional free space required in the data file	No	Yes

General Guidelines

- If an index has less than 1000 pages and is in memory, don't bother removing fragmentation
- If the index has:
  - less than 5% logical fragmentation, don't do anything
  - between 5% and 30% logical fragmentation, reorganize it (using DBCC INDEXDEFRAG or ALTER INDEX … REORGANIZE)
  - more than 30% logical fragmentation, rebuild it (using DBCC DBREINDEX or ALTER INDEX … REBUILD)

sys.dm_db_index_physical_stats DMF Example

```
SELECT OBJECT_NAME(ind.OBJECT_ID) AS TableName,
        ind.name AS IndexName,
        indexstats.index_type_desc AS IndexType,
        indexstats.avg_fragmentation_in_percent
FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, NULL)
        indexstats
INNER JOIN sys.indexes ind
ON ind.object_id = indexstats.object_id
       AND ind.index_id = indexstats.index_id
WHERE indexstats.avg_fragmentation_in_percent > 30
ORDER BY indexstats.avg_fragmentation_in_percent DESC
```