Ready-to-run T-SQL script that returns **index maintenance recommendations** (REBUILD vs REORGANIZE) **for a single database**, based on:

- **Page count > 1000**
- **avg_fragmentation_in_percent > 30 → REBUILD**
- **avg_fragmentation_in_percent between 5 and 30 → REORGANIZE**
- (optional: you may adjust thresholds)

This script uses sys.dm_db_index_physical_stats, sys.indexes, sys.objects, and includes recommended ALTER statements.

T-SQL Script: Index Fragmentation Analysis with Maintenance Suggestions

```
/*=====================================================
   Index Fragmentation Report with Maintenance Advice
   - Page count > 1000
   - Frag > 30% → REBUILD
   - Frag between 5% and 30% → REORGANIZE
=====================================================*/
            DECLARE @db_id INT = DB_ID();   -- Current database

            ;WITH FragData AS
            (
               SELECT
                   DB_NAME() AS database_name
                  , OBJECT_NAME(ips.object_id) AS table_name
                  , i.name AS index_name
                  , i.index_id
                  , ips.object_id
                  , ips.avg_fragmentation_in_percent
                  , ips.page_count
                  , ips.alloc_unit_type_desc
               FROM sys.dm_db_index_physical_stats(@db_id, NULL, NULL, NULL, 'SAMPLED') ips
               INNER JOIN sys.indexes i
                  ON ips.object_id = i.object_id
                 AND ips.index_id = i.index_id
               WHERE
                   ips.page_count > 1000
                AND i.type_desc IN ('CLUSTERED', 'NONCLUSTERED')  -- ignore heaps + XML + spatial
            )
            SELECT
                database_name,
                table_name,
                index_name,
                page_count,
                avg_fragmentation_in_percent,
                CASE
                    WHEN avg_fragmentation_in_percent >= 30 THEN 'REBUILD'
                    WHEN avg_fragmentation_in_percent BETWEEN 5 AND 30 THEN 'REORGANIZE'
                    ELSE 'NONE'
```

```
        END AS maintenance_action,
        CASE
            WHEN avg_fragmentation_in_percent >= 30
                THEN 'ALTER INDEX [' + index_name + '] ON [' + table_name + '] REBUILD WITH (ONLINE = ON);'
            WHEN avg_fragmentation_in_percent BETWEEN 5 AND 30
                THEN 'ALTER INDEX [' + index_name + '] ON [' + table_name + '] REORGANIZE;'
            ELSE '/* No Action Required */'
        END AS suggested_script
    FROM FragData
    ORDER BY avg_fragmentation_in_percent DESC;
```

📌 What This Script Does

✓ Only looks at the **current database**

✓ Uses SAMPLED mode → fast and accurate

✓ Filters to **indexes with page_count > 1000**

✓ Applies industry-recognized thresholds:

Avg Fragmentation	**Action**
≥ 30%	REBUILD (online if enterprise edition)
5–30%	REORGANIZE
< 5%	No action

✓ Produces ready-to-run **ALTER INDEX** commands

Optional: Strict fragmentation tiers

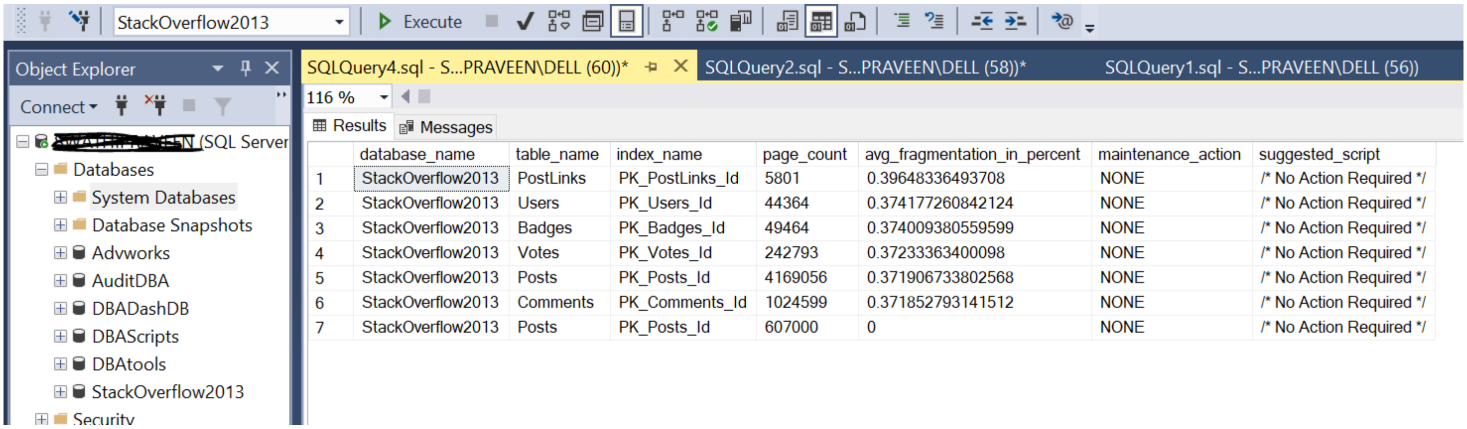
If you want only two categories (as you asked):

- **> 30% → REBUILD**
- **≤ 30% → REORGANIZE**

Use this instead:

```
CASE
    WHEN avg_fragmentation_in_percent > 30 THEN 'REBUILD'
    ELSE 'REORGANIZE'
END
```

But the industry best practice is 5–30 reorganize, <5 no action.



	database_name	table_name	index_name	page_count	avg_fragmentation_in_percent	maintenance_action	suggested_script
1	StackOverflow2013	PostLinks	PK_PostLinks_Id	5801	0.39648336493708	NONE	/* No Action Required */
2	StackOverflow2013	Users	PK_Users_Id	44364	0.374177260842124	NONE	/* No Action Required */
3	StackOverflow2013	Badges	PK_Badges_Id	49464	0.374009380559599	NONE	/* No Action Required */
4	StackOverflow2013	Votes	PK_Votes_Id	242793	0.37233363400098	NONE	/* No Action Required */
5	StackOverflow2013	Posts	PK_Posts_Id	4169056	0.371906733802568	NONE	/* No Action Required */
6	StackOverflow2013	Comments	PK_Comments_Id	1024599	0.371852793141512	NONE	/* No Action Required */
7	StackOverflow2013	Posts	PK_Posts_Id	607000	0	NONE	/* No Action Required */

Production-ready script that runs fragmentation analysis **for all user DB's only** (excludes master, model, msdb, tempdb).

It dynamically loops through user DBs, executes dm_db_index_physical_stats in each context, aggregates results, and provides **REBUILD/REORGANIZE recommendations**.

T-SQL: Fragmentation Report for ALL User Databases

```
/*=====================================================
  Index Fragmentation for ALL USER Databases
  - Page count > 1000
  - Frag > 30% → REBUILD
  - Frag 5–30% → REORGANIZE
  - Excludes system databases
=====================================================*/

IF OBJECT_ID('tempdb..#Frag') IS NOT NULL DROP TABLE #Frag;

CREATE TABLE #Frag
(
    database_name sysname,
    schema_name sysname,
    table_name sysname,
    index_name sysname,
    index_id int,
    object_id int,
    avg_fragmentation_in_percent float,
    page_count bigint
);

DECLARE @db sysname;
DECLARE @sql NVARCHAR(MAX);

DECLARE db_cursor CURSOR LOCAL FAST_FORWARD FOR
SELECT name
FROM sys.databases
WHERE database_id > 4      -- user DBs only
  AND state_desc = 'ONLINE';

OPEN db_cursor;
FETCH NEXT FROM db_cursor INTO @db;

WHILE @@FETCH_STATUS = 0
BEGIN
    SET @sql = '
        USE ' + QUOTENAME(@db) + ';
        INSERT INTO #Frag
        SELECT
            DB_NAME() AS database_name,
            SCHEMA_NAME(o.schema_id) AS schema_name,
            o.name AS table_name,
      i.name AS index_name,
      i.index_id,
      i.object_id,
      ips.avg_fragmentation_in_percent,
      ips.page_count
    FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, ''SAMPLED'') ips
    JOIN sys.indexes i ON ips.object_id = i.object_id AND ips.index_id = i.index_id
    JOIN sys.objects o ON o.object_id = ips.object_id
    WHERE ips.page_count > 1000
     AND i.type_desc IN (''CLUSTERED'', ''NONCLUSTERED'')
     AND o.is_ms_shipped = 0;
  ';

  EXEC sys.sp_executesql @sql;
  FETCH NEXT FROM db_cursor INTO @db;
END

CLOSE db_cursor;
DEALLOCATE db_cursor;

SELECT *,
    CASE WHEN avg_fragmentation_in_percent >= 30 THEN 'REBUILD'
      WHEN avg_fragmentation_in_percent BETWEEN 5 AND 30 THEN 'REORGANIZE'
      ELSE 'NONE'
    END AS maintenance_action
FROM #Frag
ORDER BY avg_fragmentation_in_percent DESC;
```

📌 What This Script Does

✓ Loops through **user databases only**

✓ Runs sys.dm_db_index_physical_stats in each DB

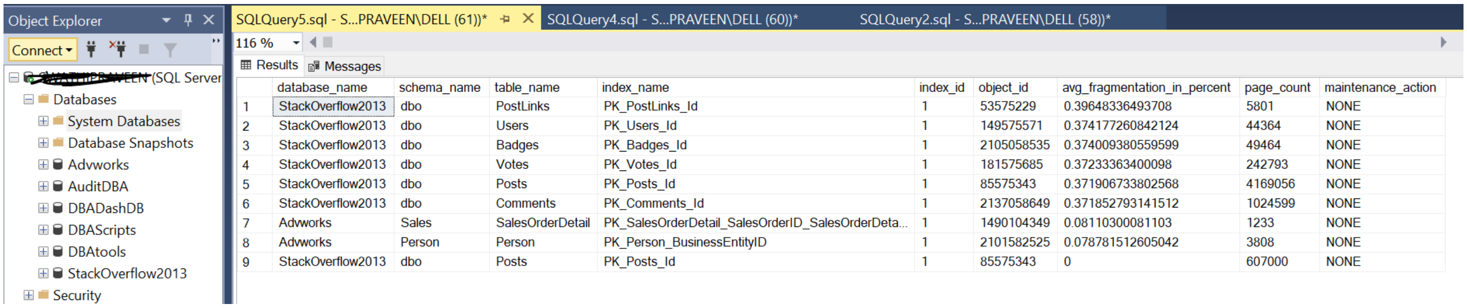
✓ Collects fragmentation data into a temp table

✓ Filters out:

- Page count ≤ 1000
- Non-clustered / clustered only
- System databases

✓ Produces **REBUILD / REORGANIZE / NONE** recommendation

✓ Generates ready-to-run ALTER INDEX commands



	database_name	schema_name	table_name	index_name	index_id	object_id	avg_fragmentation_in_percent	page_count	maintenance_action
1	StackOverflow2013	dbo	PostLinks	PK_PostLinks_Id	1	53575229	0.39648336493708	5801	NONE
2	StackOverflow2013	dbo	Users	PK_Users_Id	1	149575571	0.374177260842124	44364	NONE
3	StackOverflow2013	dbo	Badges	PK_Badges_Id	1	2105058535	0.374009380559599	49464	NONE
4	StackOverflow2013	dbo	Votes	PK_Votes_Id	1	181575685	0.37233363400098	242793	NONE
5	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0.371906733802568	4169056	NONE
6	StackOverflow2013	dbo	Comments	PK_Comments_Id	1	2137058649	0.371852793141512	1024599	NONE
7	Advworks	Sales	SalesOrderDetail	PK_SalesOrderDetail_SalesOrderID_SalesOrderDeta...	1	1490104349	0.08110300081103	1233	NONE
8	Advworks	Person	Person	PK_Person_BusinessEntityID	1	2101582525	0.078781512605042	3808	NONE
9	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0	607000	NONE

1. Version that saves results to a permanent history table

Creates a table and stores every execution run for long-term tracking.

Step 1 — Create History Table

```
USE DBA;   -- change to your DBA/Utility database

IF OBJECT_ID('DBA.dbo.IndexFragmentationHistory') IS NULL
BEGIN
   CREATE TABLE DBA.dbo.IndexFragmentationHistory
   (
      RunID INT IDENTITY(1,1) PRIMARY KEY,
      RunDate DATETIME DEFAULT GETDATE(),
      database_name SYSNAME,
      schema_name SYSNAME,
      table_name SYSNAME,
      index_name SYSNAME,
      index_id INT,
      object_id INT,
      avg_fragmentation_in_percent FLOAT,
      page_count BIGINT,
      maintenance_action NVARCHAR(20),
      suggested_script NVARCHAR(MAX)
   );
END
```

Step 2 — Insert Results for ALL User Databases

```
/*========================================================
  All User DB Fragmentation → Save to History Table
========================================================*/

-- Temporary table to collect fragmentation
IF OBJECT_ID('tempdb..#Frag') IS NOT NULL DROP TABLE #Frag;

CREATE TABLE #Frag
(
   database_name SYSNAME,
   schema_name SYSNAME,
   table_name SYSNAME,
   index_name SYSNAME,
   index_id INT,
   object_id INT,
   avg_fragmentation_in_percent FLOAT,
   page_count BIGINT,
   alloc_unit_type_desc NVARCHAR(60)
);
```

```
DECLARE @db SYSNAME;
DECLARE @sql NVARCHAR(MAX);

DECLARE db_cursor CURSOR LOCAL FAST_FORWARD FOR
SELECT name
FROM sys.databases
WHERE database_id > 4
  AND state_desc = 'ONLINE'
  AND user_access_desc = 'MULTI_USER'
ORDER BY name;

OPEN db_cursor;
FETCH NEXT FROM db_cursor INTO @db;

WHILE @@FETCH_STATUS = 0
BEGIN
    SET @sql = N'
    INSERT INTO #Frag
    SELECT
          DB_NAME() AS database_name,
          s.name AS schema_name,
          o.name AS table_name,
          i.name AS index_name,
          i.index_id,
          ips.object_id,
          ips.avg_fragmentation_in_percent,
          ips.page_count,
          ips.alloc_unit_type_desc
    FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, ''SAMPLED'') ips
    INNER JOIN sys.indexes i
            ON ips.object_id = i.object_id
           AND ips.index_id = i.index_id
    INNER JOIN sys.objects o
            ON o.object_id = i.object_id
    INNER JOIN sys.schemas s
            ON s.schema_id = o.schema_id
    WHERE ips.page_count > 1000
      AND i.type_desc IN (''CLUSTERED'', ''NONCLUSTERED'')
      AND o.is_ms_shipped = 0;
    ';

    -- Switch context safely
    SET @sql = N'USE ' + QUOTENAME(@db) + '; ' + @sql;

    EXEC sp_executesql @sql;
```

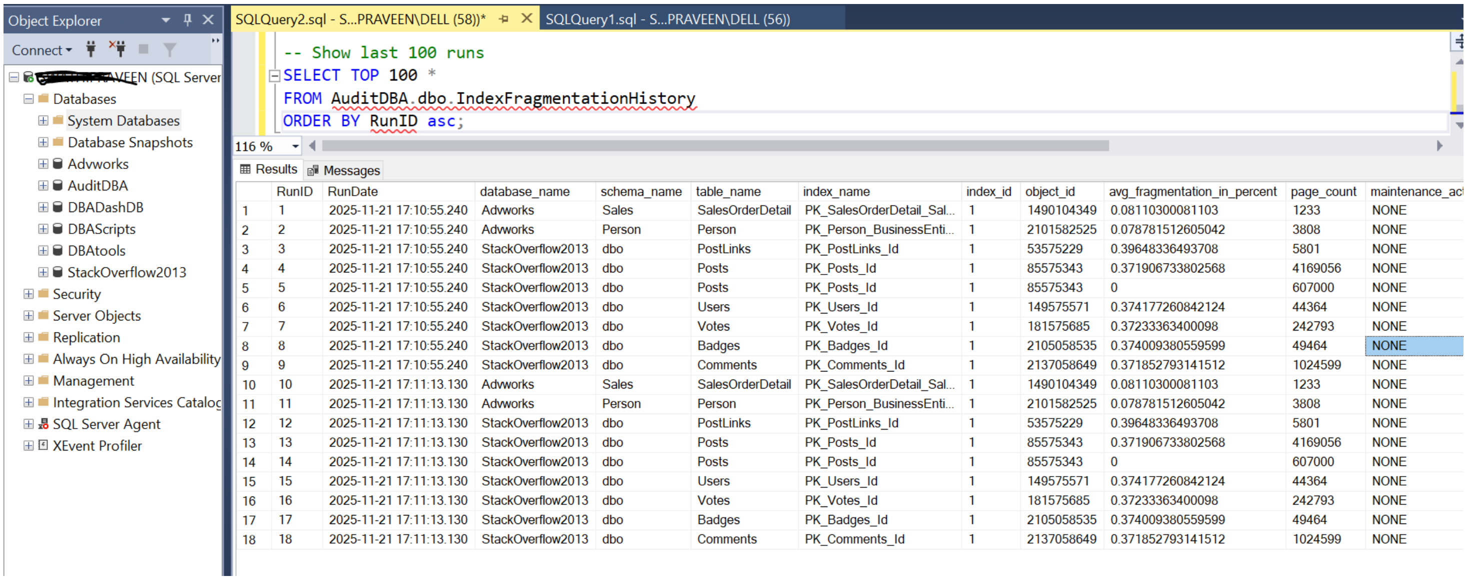
```
      FETCH NEXT FROM db_cursor INTO @db;
    END

    CLOSE db_cursor;
    DEALLOCATE db_cursor;

    -- Insert into history table
    INSERT INTO DBA.dbo.IndexFragmentationHistory
    (
      database_name, schema_name, table_name, index_name, index_id,
      object_id, avg_fragmentation_in_percent, page_count,
      maintenance_action, suggested_script
    )
    SELECT
        database_name,
        schema_name,
        table_name,
        index_name,
        index_id,
        object_id,
        avg_fragmentation_in_percent,
        page_count,
        CASE
            WHEN avg_fragmentation_in_percent >= 30 THEN 'REBUILD'
            WHEN avg_fragmentation_in_percent BETWEEN 5 AND 30 THEN 'REORGANIZE'
            ELSE 'NONE'
        END AS maintenance_action,
        CASE
            WHEN avg_fragmentation_in_percent >= 30
                THEN 'ALTER INDEX ' + QUOTENAME(index_name) + ' ON ' + QUOTENAME(schema_name) + '.' +
    QUOTENAME(table_name) + ' REBUILD WITH (ONLINE = ON);'
            WHEN avg_fragmentation_in_percent BETWEEN 5 AND 30
                THEN 'ALTER INDEX ' + QUOTENAME(index_name) + ' ON ' + QUOTENAME(schema_name) + '.' +
    QUOTENAME(table_name) + ' REORGANIZE;'
            ELSE '/* No Action Required */'
        END AS suggested_script
    FROM #Frag;

-- Show last 100 runs
    SELECT TOP 100 *
    FROM DBA.dbo.IndexFragmentationHistory
    ORDER BY RunID ASC;
```

✅ 2. Version that EXCLUDES specific DBs (e.g., ReportServer, AuditDB)

Modify the cursor query:

```
WHERE database_id > 4
AND state_desc = 'ONLINE'
AND name NOT IN ('ReportServer', 'ReportServerTempDB', 'AuditDB')
```

Sample output:



```
-- Show last 100 runs
SELECT TOP 100 *
FROM AuditDBA.dbo.IndexFragmentationHistory
ORDER BY RunID asc;
```

	RunID	RunDate	database_name	schema_name	table_name	index_name	index_id	object_id	avg_fragmentation_in_percent	page_count	maintenance_act
1	1	2025-11-21 17:10:55.240	Advworks	Sales	SalesOrderDetail	PK_SalesOrderDetail_Sal...	1	1490104349	0.08110300081103	1233	NONE
2	2	2025-11-21 17:10:55.240	Advworks	Person	Person	PK_Person_BusinessEnti...	1	2101582525	0.078781512605042	3808	NONE
3	3	2025-11-21 17:10:55.240	StackOverflow2013	dbo	PostLinks	PK_PostLinks_Id	1	53575229	0.39648336493708	5801	NONE
4	4	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0.371906733802568	4169056	NONE
5	5	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0	607000	NONE
6	6	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Users	PK_Users_Id	1	149575571	0.374177260842124	44364	NONE
7	7	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Votes	PK_Votes_Id	1	181575685	0.37233363400098	242793	NONE
8	8	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Badges	PK_Badges_Id	1	2105058535	0.374009380559599	49464	NONE
9	9	2025-11-21 17:10:55.240	StackOverflow2013	dbo	Comments	PK_Comments_Id	1	2137058649	0.371852793141512	1024599	NONE
10	10	2025-11-21 17:11:13.130	Advworks	Sales	SalesOrderDetail	PK_SalesOrderDetail_Sal...	1	1490104349	0.08110300081103	1233	NONE
11	11	2025-11-21 17:11:13.130	Advworks	Person	Person	PK_Person_BusinessEnti...	1	2101582525	0.078781512605042	3808	NONE
12	12	2025-11-21 17:11:13.130	StackOverflow2013	dbo	PostLinks	PK_PostLinks_Id	1	53575229	0.39648336493708	5801	NONE
13	13	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0.371906733802568	4169056	NONE
14	14	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Posts	PK_Posts_Id	1	85575343	0	607000	NONE
15	15	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Users	PK_Users_Id	1	149575571	0.374177260842124	44364	NONE
16	16	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Votes	PK_Votes_Id	1	181575685	0.37233363400098	242793	NONE
17	17	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Badges	PK_Badges_Id	1	2105058535	0.374009380559599	49464	NONE
18	18	2025-11-21 17:11:13.130	StackOverflow2013	dbo	Comments	PK_Comments_Id	1	2137058649	0.371852793141512	1024599	NONE