Level Up Your SQL Server Index Knowledge

Jes Borland, Senior SQL Engineer

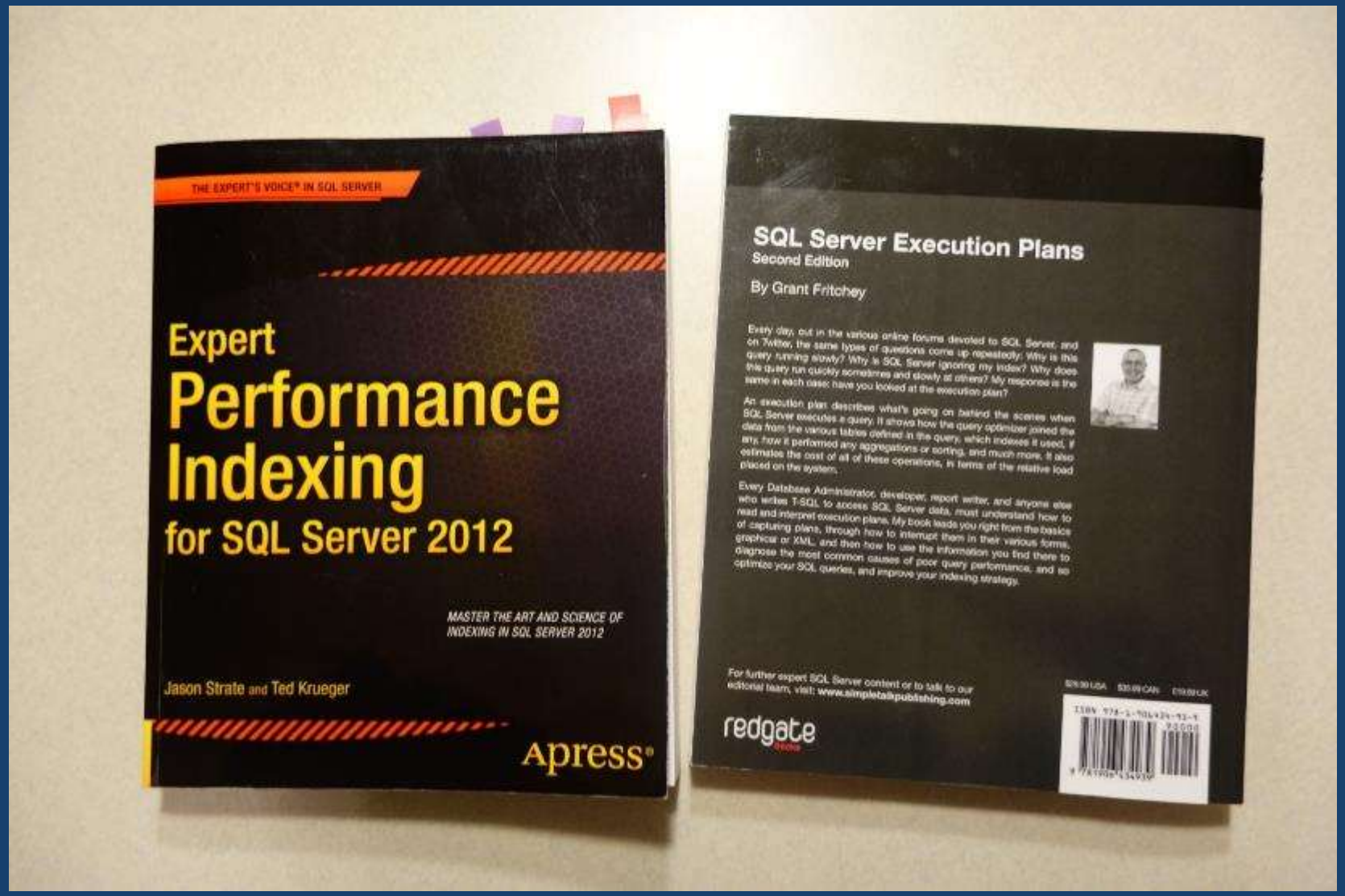
Yes!

- Slides and code will be posted to my blog
- http://lessthandot.com

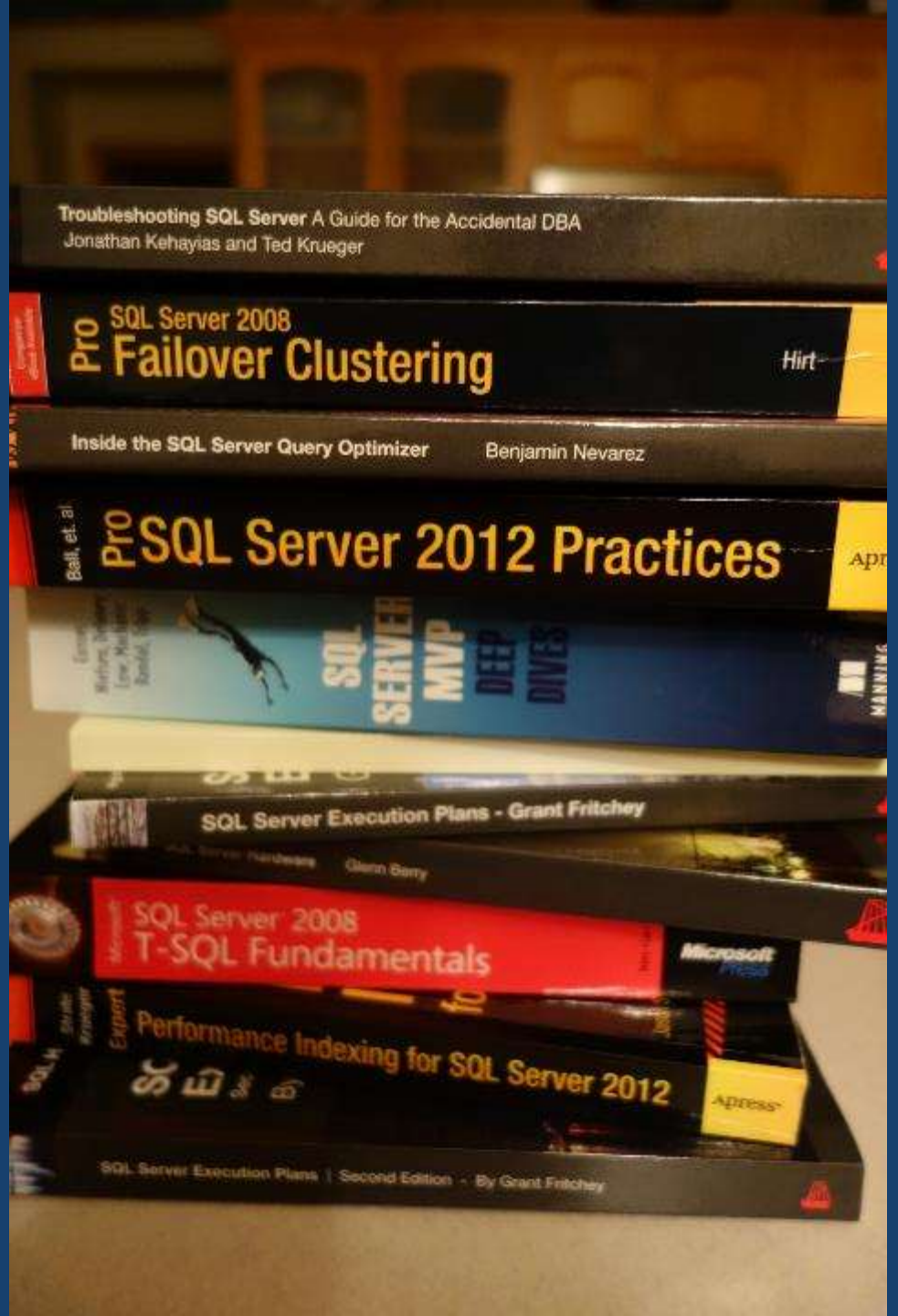
Level 1: Heaps, Clusters, and Primary Keys

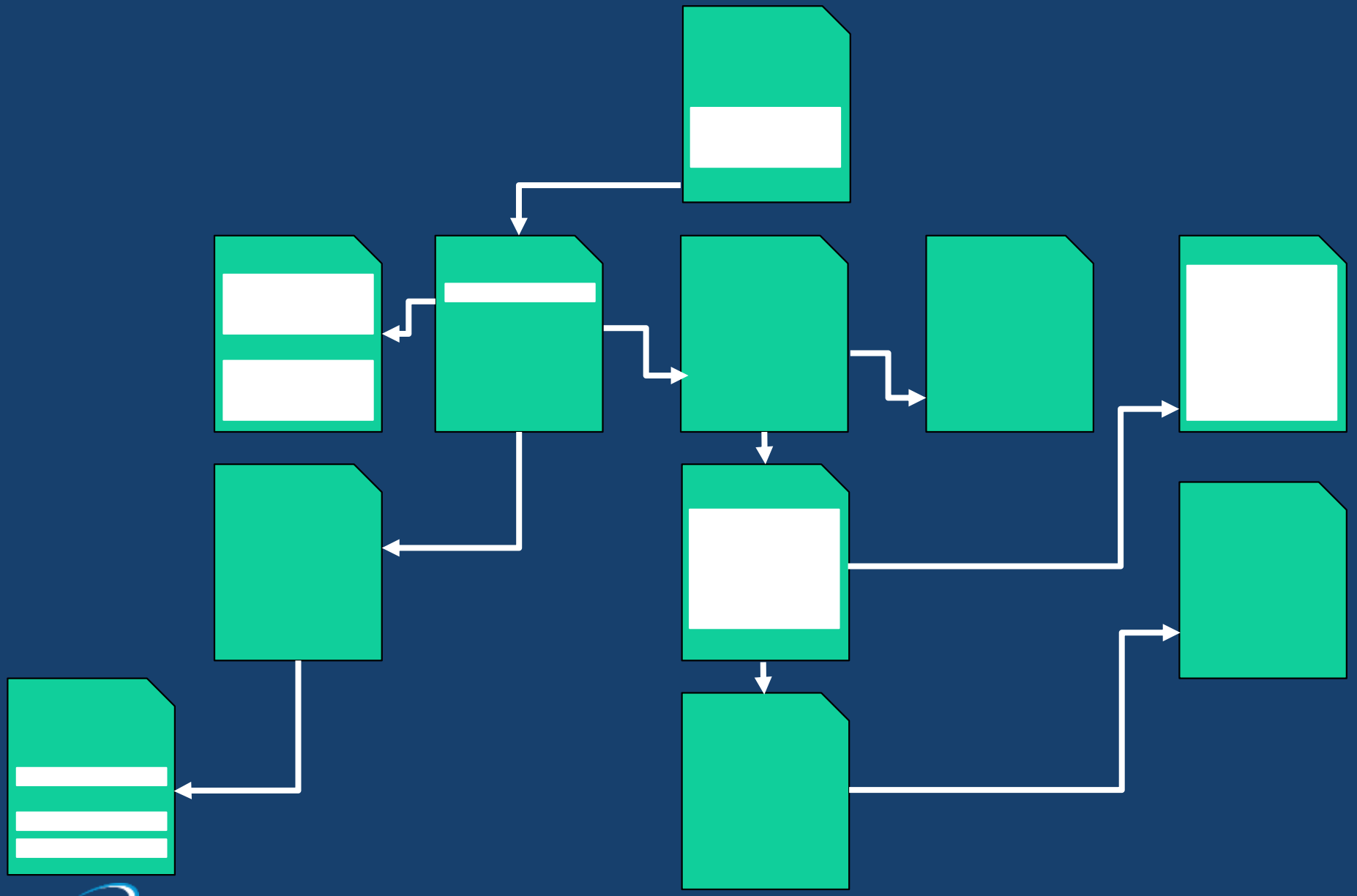
Books

- Title
- Authors ID
- Publisher ID
- Published Year
- ISBN10
- ISBN13



Heap





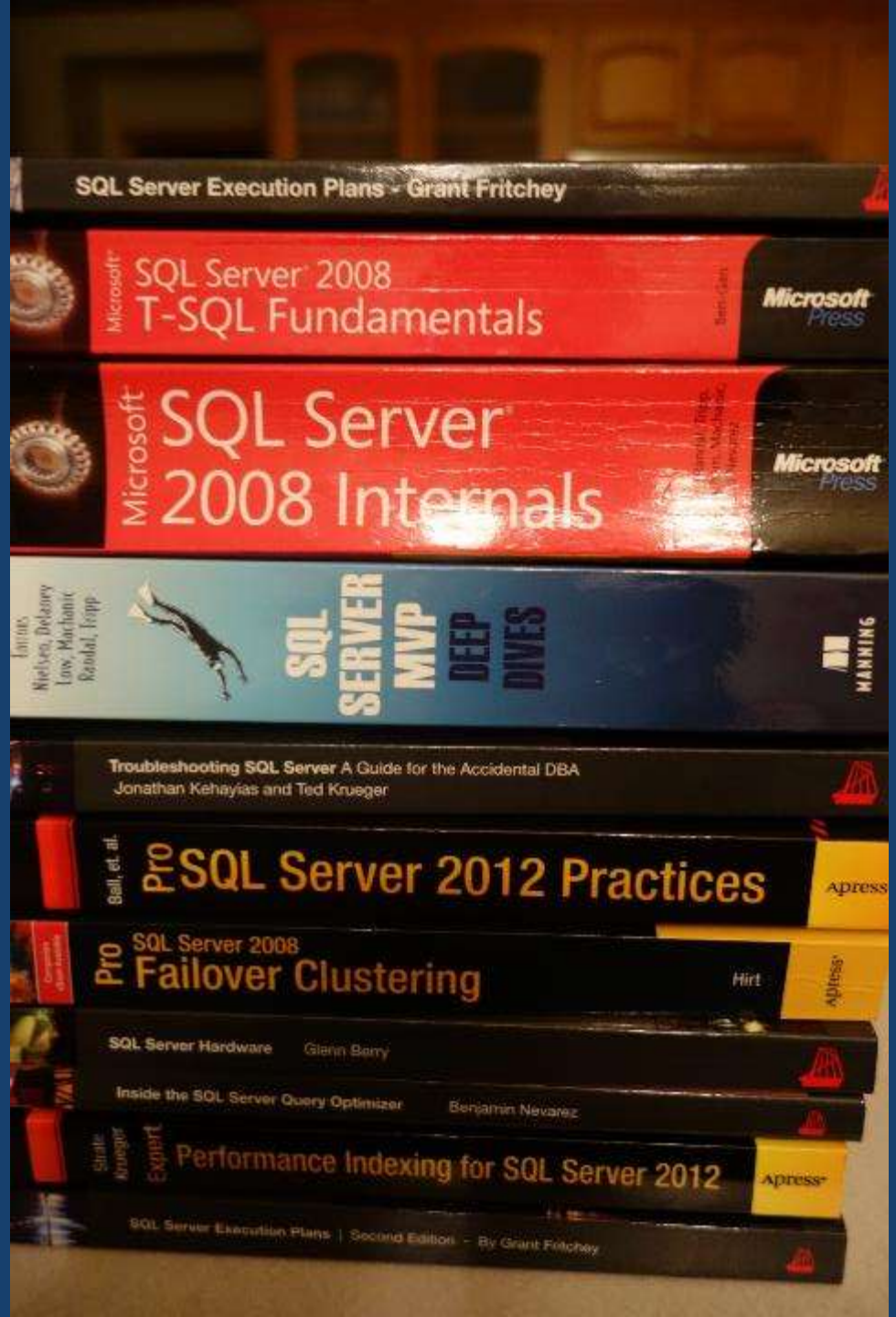
What's stored on the pages

Title	AuthorsID	PublisherID	PublishedYear	ISBN10	ISBN13	RID
Troubleshooting SQL Server	7	1004	2011	1906434786	978-1906434786	1:16:2
Expert Performance Indexing	9	2396	2012	1430237414	978-1430237419	1:38:9
SQL Server MVP Deep Dives	87	987	2010	1935182048	978-1935182047	1:11:1
SQL Server 2008 Internals	16	1543	2009	0735626243	978-0735626249	1:78:23
Inside the SQL Server Query Optimizer	78	1004	2011	1906434603	978-1906434601	1:25:4

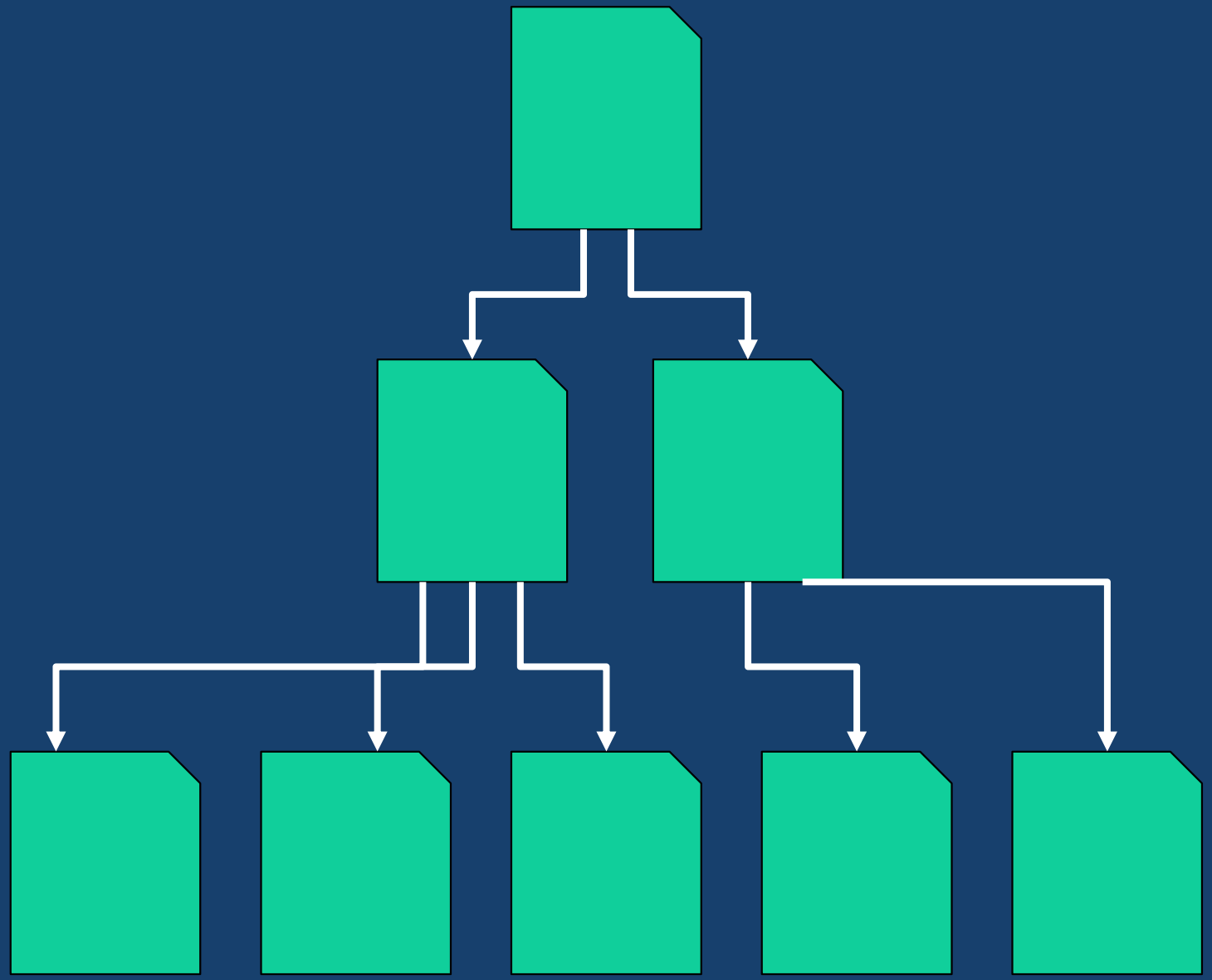
Row identifier

- Auto-assigned number that uniquely identifies each row in the heap
- A pointer to its physical location

Clustered index



Concurrency



What's stored on the "leaf" pages

BookID	Title	AuthorsID	PublisherID	PublishedYear	ISBN10	ISBN13
1001	Troubleshooting SQL Server	7	1004	2011	1906434786	978-1906434786
1002	Expert Performance Indexing	9	2396	2012	1430237414	978-1430237419
1003	SQL Server MVP Deep Dives	87	987	2010	1935182048	978-1935182047
1004	SQL Server 2008 Internals	16	1543	2009	0735626243	978-0735626249
1006	Inside the SQL Server Query Optimizer	78	1004	2011	1906434603	978-1906434601

What makes a good clustered index key?

- Sequential
- Ever-increasing
- Low cardinality

What's a Primary Key?

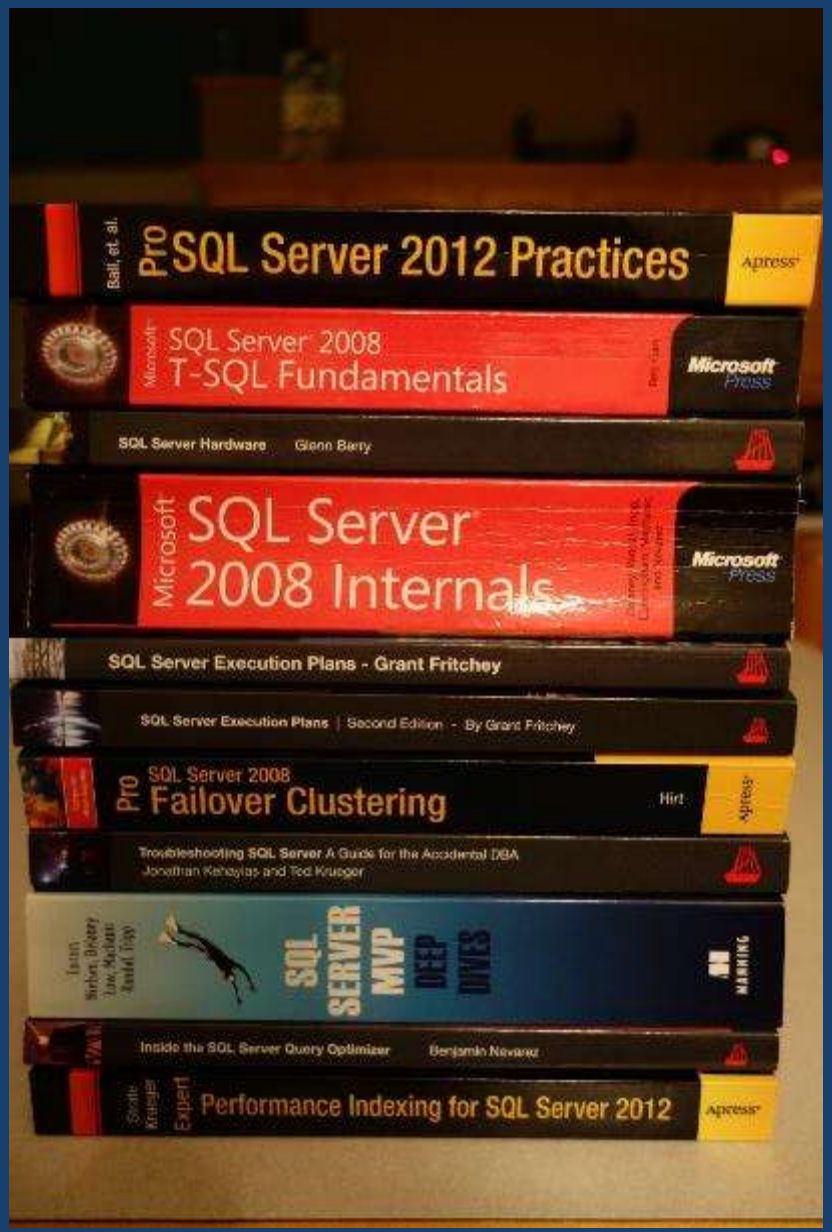
- Unique!
- Relational
  - Foreign Key constraints
- Required to implement some features in SQL Server
  - Transactional Replication
  - Change Data Capture
  - Auditing

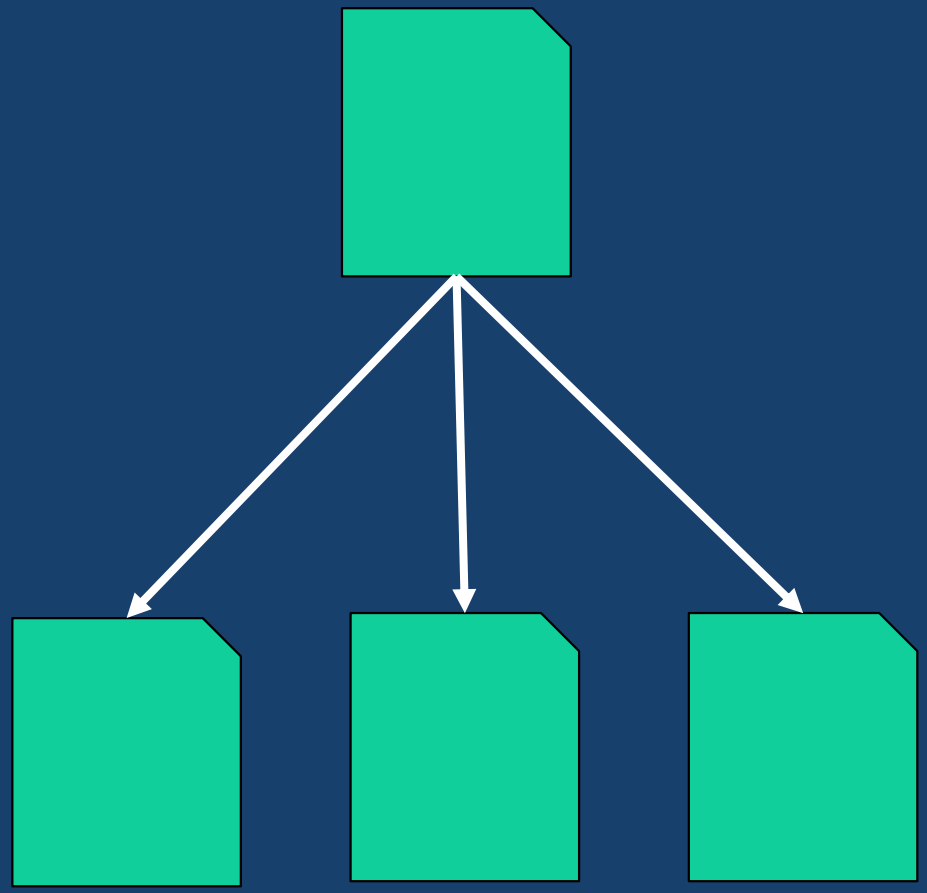
Should I use the clustered index as my primary key?

- Would the columns that make each record unique be
  - Sequential?
  - Ever-increasing?
  - Low cardinality?

The clustered index is a technical definition,
the primary key is a business definition

Level 2: Nonclustered indexes





What's stored on the "leaf" pages

Title

Title	BookID
Expert Performance Indexing	1002
Inside the SQL Server Query Optimizer	1006
SQL Server 2008 Internals	1004
SQL Server MVP Deep Dives	1003
Troubleshooting SQL Server	1001

Author

AuthorsID	BookID
7	1001
9	1002
16	1004
78	1006
87	1003

Publisher

PublisherID	BookID
987	1003
1004	1001
1004	1006
1543	1004
2396	1002

Storing the row identifier or clustered index key

- Every row
- Points back to heap or clustered index

Key vs include columns

```
CREATE NONCLUSTERED
INDEX
IX_Books_ISBN10
(ISBN10)
```

ISBN10	BookID
0735626243	1004
1430237414	1002
1906434603	1006
1906434786	1001
1935182048	1003

```
SELECT Title,
ISBN10, ISBN13
FROM Books
WHERE ISBN10 =
@ISBN10
```

With INCLUDE

```
CREATE NONCLUSTERED INDEX
IX_Books_ISBN10 (ISBN10)
INCLUDE (Title, ISBN13)
```

ISBN10	Title	ISBN13	BookID
0735626243	SQL Server 2008 Internals	978-0735626249	1004
1430237414	Expert Performance Indexing	978-1430237419	1002
1906434603	Inside the SQL Server Query Optimizer	978-1906434601	1006
1906434786	Troubleshooting SQL Server	978-1906434786	1001
1935182048	SQL Server MVP Deep Dives	978-1935182047	1003

```
SELECT Title, ISBN10,
ISBN13
FROM Books
WHERE ISBN10 =
@ISBN10
```

Key columns support WHERE, ORDER BY
Include columns support SELECT

Filtered nonclustered indexes

ISBN13

- 10 million records
- 1 million do not have an ISBN13

```
SELECT Title, ISBN10, ISBN13
FROM Books
WHERE ISBN13 IS NULL
```

Index already exists

```
CREATE NONCLUSTERED INDEX
IX_Books_ISBN13 (ISBN13)
```

- But…
- All 10 million records are scanned

Alternative

```
CREATE NONCLUSTERED INDEX
IX_Books_ISBN13_FILTERED (ISBN13)
WHERE ISBN13 IS NULL
```

- Only the 1 million records WHERE ISBN13 IS NULL are stored
- Thus, only 1 million would be scanned
- 90% reduction in reads!

Concurrency

Rules

- Use an equality or inequality operator such as =, >, <, >=, <=, IN
- Don't use BETWEEN, NOT IN, CASE
- The expression must be deterministic

The Joys of Filtered Indexes:
http://blogs.msdn.com/b/timchapman/archive/2012/08/27/the-joys-of-filtered-indexes.aspx
The Pains of Filtered Indexes:
http://blogs.msdn.com/b/timchapman/archive/2012/08/27/the-drawback-of-using-filtered-indexes.aspx

Level 3:
Use SQL Server's index DMVs for more insight

Questions we can answer

- What indexes exist?
- How many times have they been read from or written to?
- What missing indexes does SQL Server want me to add?

Find existing indexes

sys.indexes

- Name
- Type
- If unique
- If primary key
- If filtered
- Column names
- Column data types

Results

TableName	IndexName	TypeOfIndex	IsUnique	IsPrimaryKey	IsFilteredIndex	ColumnName	DataType	ColMaxLength	ColOrder
dbo.AWBuildVersion	PK_AWBuildVersion_SystemInformationID	CLUSTERED	Yes	Yes		SystemInformationID	tinyint	1	ASC
dbo.bigProduct	IX_bigProduct_Color	NONCLUSTERED				Color	nvarchar	30	ASC
dbo.bigProduct	IX_bigProduct_Name	NONCLUSTERED				Name	nvarchar	160	ASC
dbo.bigProduct	IX_bigProduct_Size	NONCLUSTERED				Size	nvarchar	10	ASC
dbo.bigProduct	IX_bigProduct_Weight	NONCLUSTERED				Weight	decimal	5	ASC
dbo.bigProduct	pk_bigProduct	CLUSTERED	Yes	Yes		ProductID	int	4	ASC
dbo.bigTransactionHistory	pk_bigTransactionHistory	CLUSTERED	Yes	Yes		TransactionID	int	4	ASC
dbo.DatabaseLog	PK_DatabaseLog_DatabaseLogID	NONCLUSTERED	Yes	Yes		DatabaseLogID	int	4	ASC
dbo.DeadlockTest	CX_DeadlockTest	CLUSTERED				ID	int	4	ASC
dbo.ErrorLog	PK_ErrorLog_ErrorLogID	CLUSTERED	Yes	Yes		ErrorLogID	int	4	ASC
dbo.ProductInventoryByLocation	CX_ProductInventoryByLocation	CLUSTERED	Yes			ProductID	int	4	ASC
dbo.ProductInventoryByLocation	CX_ProductInventoryByLocation	CLUSTERED	Yes			LocationID	smallint	2	ASC

Index usage

sys.dm_db_index_usage_stats

- User reads & writes
- System reads & writes

Results

database_name	table_name	index_name	user_seeks	user_scans	user_lookups	user_updates	system_seeks	system_scans	system_lookups	system_updates
msdb	JobCandidate	IX_JobCandidate_BusinessEntityID	0	1	0	0	0	0	0	0
msdb	JobCandidate	PK_JobCandidate_JobCandidateID	8	1	0	8	0	0	0	0
msdb	Employee	PK_Employee_BusinessEntityID	2	1	0	0	0	0	0	0
msdb	CurrencyRate	PK_CurrencyRate_CurrencyRateID	0	1	0	12	0	0	0	0
msdb	CurrencyRate	AK_CurrencyRate_CurrencyRateDate_FromCurrencyCode_ToCurrencyCode	0	1	0	12	0	0	0	0
AdventureWorks2012	Person	IX_Person_LastName_FirstName_MiddleName	0	1	0	0	0	0	0	0

What are we really looking for here?

- Unused indexes
- Under-used indexes

Missing indexes

- sys.dm_db_missing_index_details
- sys.dm_db_missing_index_groups
- sys.dm_db_missing_index_group_stats

Results

table_name	user_seeks	user_scans	avg_total_user_cost	avg_user_impact	equality_columns	inequality_columns	included_columns
[AdventureWorks2012].[Sales].[SalesOrderHeader]	2	0	0.559663	79.85	[Status]	NULL	[OrderDate], [DueDate], [ShipDate]

Not a panacea

- Is it similar to an already-existing index that could be modified?
- How many times would it be read from, compared to written to?
- It doesn't give you the create index statement
  - Key columns = equality_columns + inequality_columns, most cardinal first
  - INCLUDE = included_columns
- DMV is reset when SQL Server is restarted
- Only 500 missing indexes will be kept in the DMV

Questions the DMVs can't answer

- Should I add the missing index?
- Is that under-used index important?
- How many nonclustered indexes is too many for this table?

Level 4: Index resources

Books, Blogs, Courses

- Expert Performance Indexing for SQL Server 2012 http://www.apress.com/9781430237419?gtmf=s
- Stairway to SQL Server Indexes http://www.sqlservercentral.com/stairway/72399/
- SQL Server Performance: Indexing Basics http://www.pluralsight.com/courses/sql-server-indexing

Level 5: Questions? Reach out!

- jborland@concurrency.com
- http://lessthandot.com
- @grrl_geek