PASS24
Data Community Summit

Mastering SQL Server's Core: A Deep Dive into Internals

John Morehouse

He/Him

Principal Consultant

Denny Cherry & Associates Consulting

John

Morehouse

He/Him

Principal Consultant

Denny Cherry & Associates Consulting

I like solving business critical needs utilizing the SQL Server platform, regardless if that's on-premises or in the cloud. Also, I'm a nerd.

Currently, I'm working at Denny Cherry & Associates Consulting, a company based in California, USA.

@SQLRUS

https://www.sqlrus.com

https://linkedin.com/in/johnmorehouse

Denny Cherry & Associates

Certified IT professionals to help achieve IT goals

Clients ranging from small business to Fortune 10 corporations

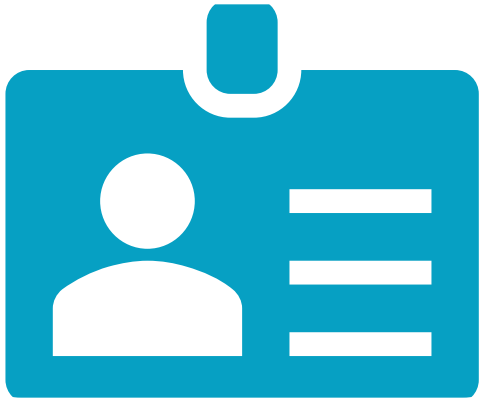
Help save on costs while improving IT reliability and solving challenges

DCAC

My Slides & Demos

https://bit.ly/mypresentationfiles

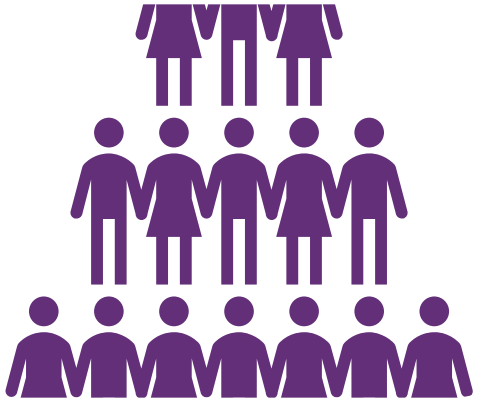
About You



DBAs, Devs, Managers, BI,

Others

Years of Experience



Levels Newbie, Junior, Senior, Principals

Has been in this session before

Agenda

Why it matters

Records & Pages

Demos

Break

Agenda

Log files

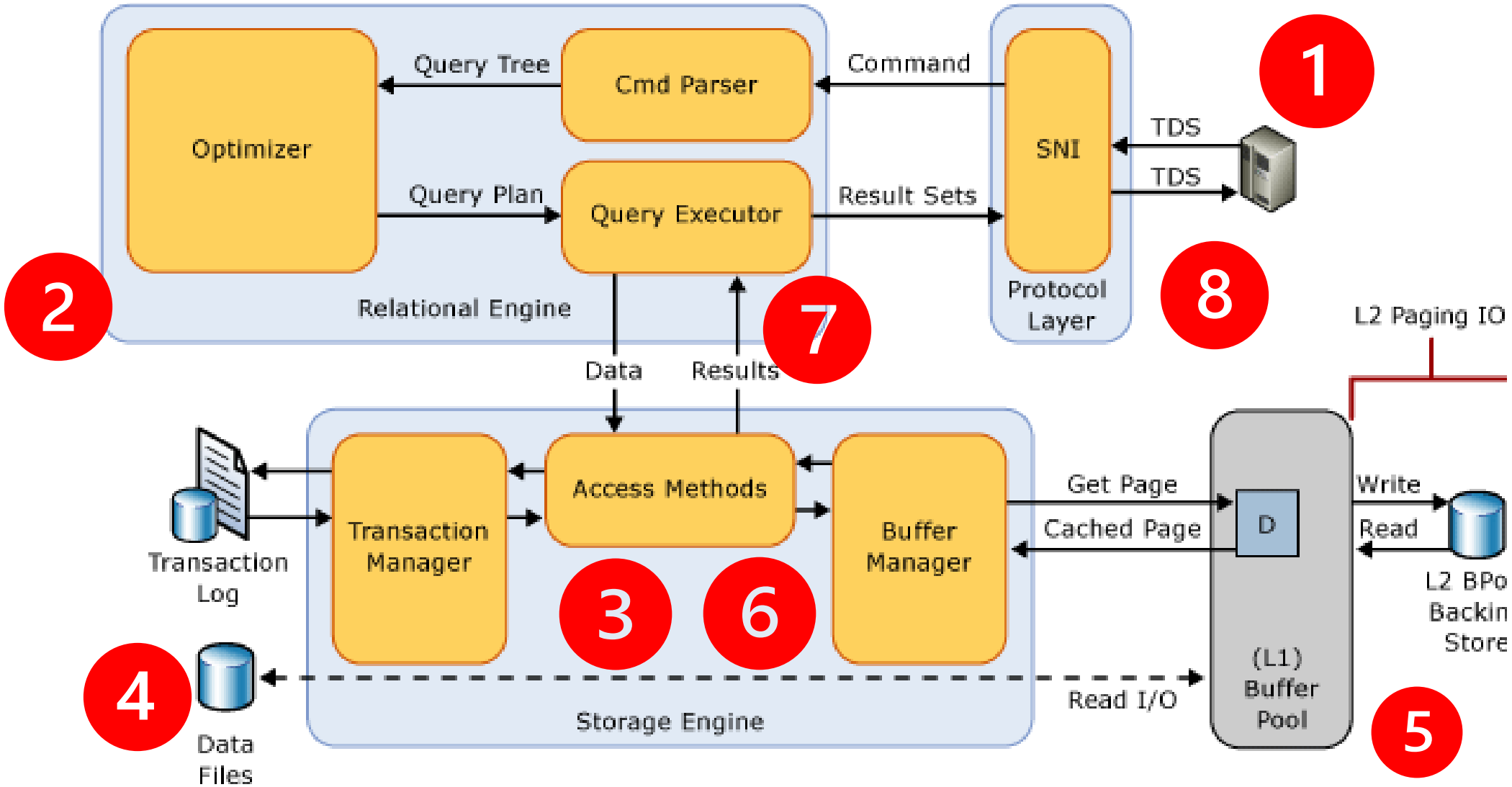
Demos

Q&A

Summary


Why do I even care?

ENGINE ARCHITECTURE



Source: https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/buffer-pool-extension?view=sql-server-ver15

Foundations

Overall Structure

Records = Rows = Slots

Records live in Pages

Groups of 8 pages is an Extent

Record contents

- Data
- Forwarding
- Index
- Versioned Records
- Ghost Records
- Large Object (LOB)
- And so on…

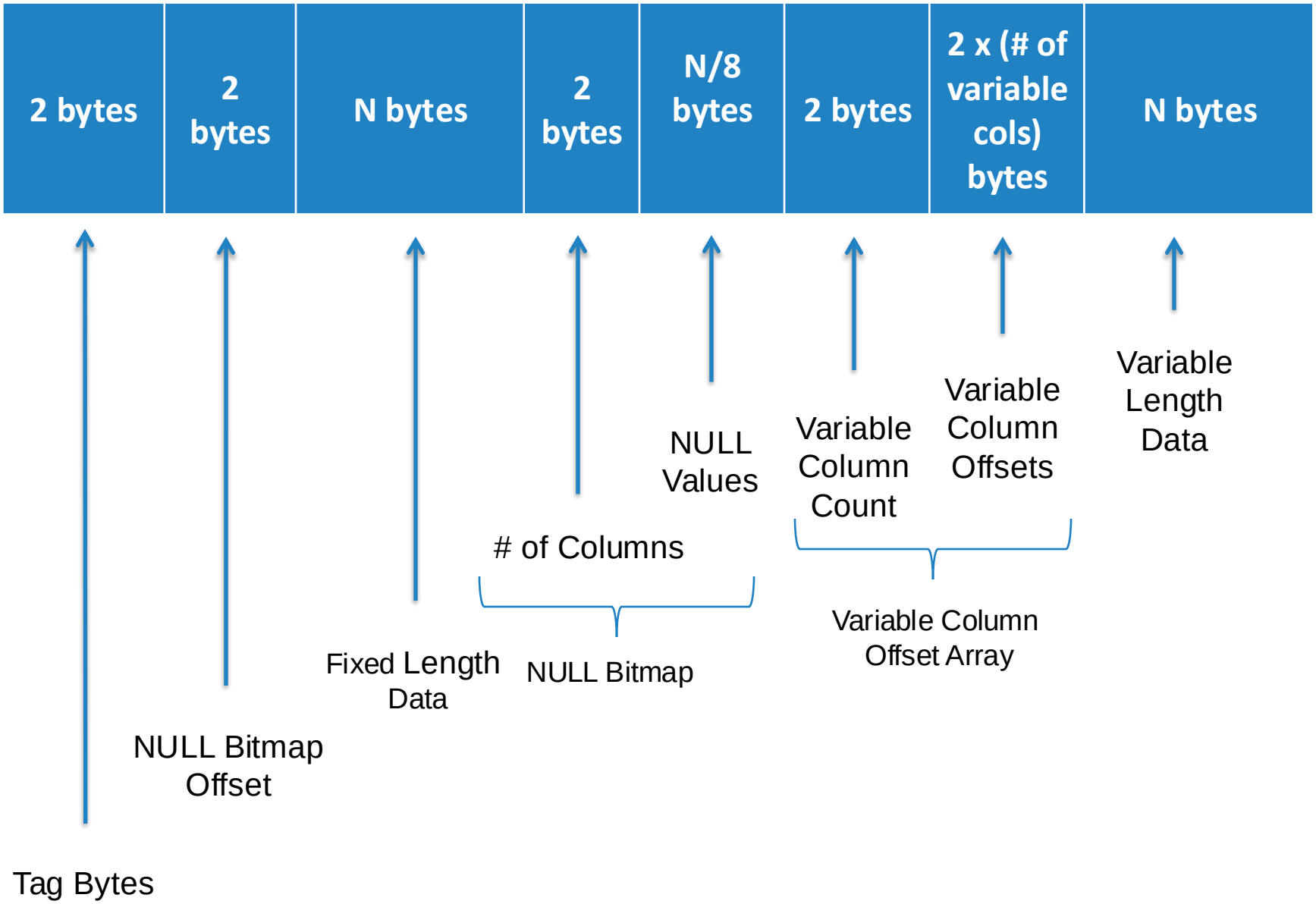
All data stored is either Fixed Length or Variable Length

Data Types Review

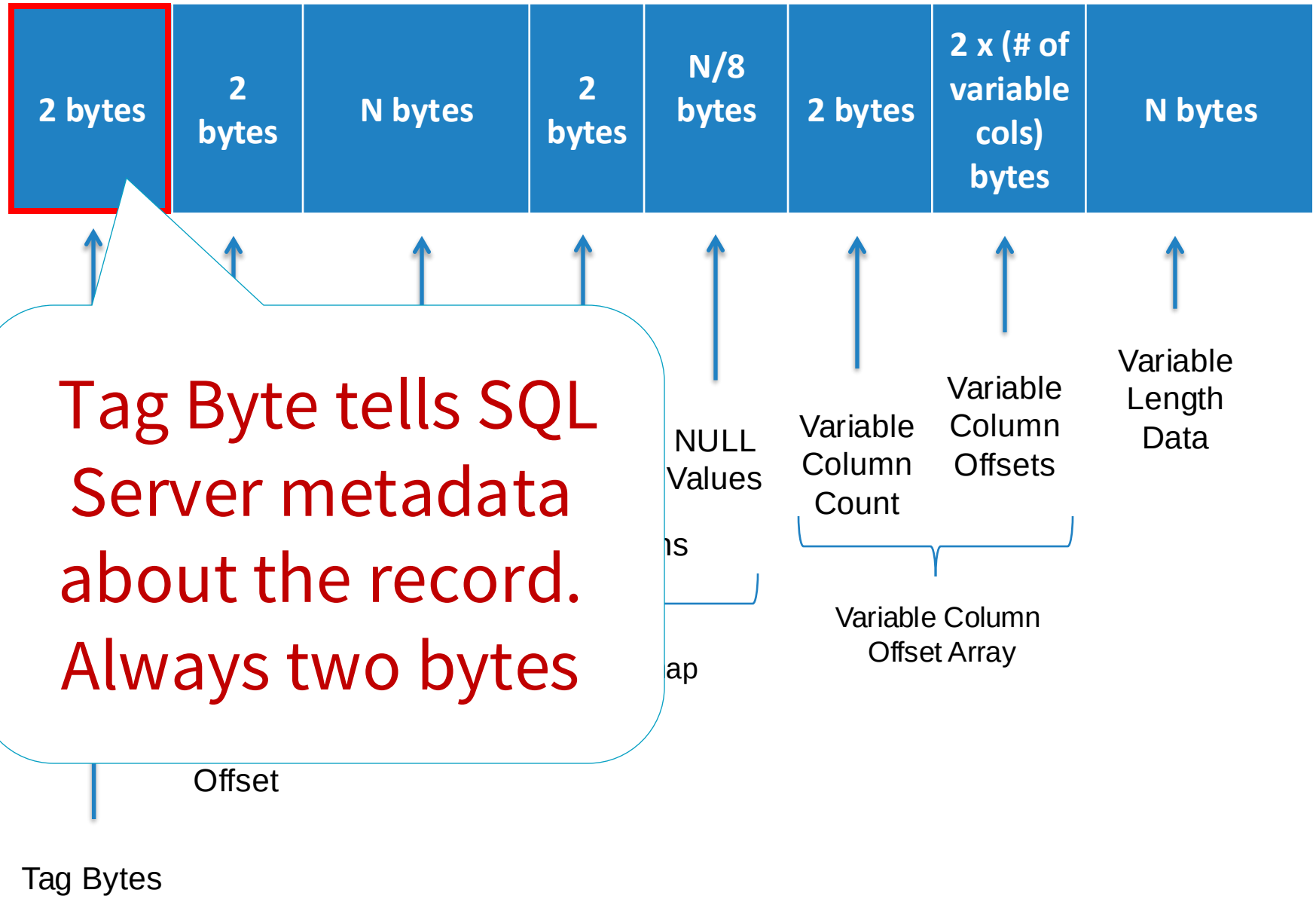
Type	Size	Max Size
INTEGER	4 bytes	
DATETIME	8 bytes	
CHAR	X bytes	
NCHAR	2*X bytes	
VARCHAR	X bytes	2GB
NVARCHAR	2*X bytes	2GB

Rows

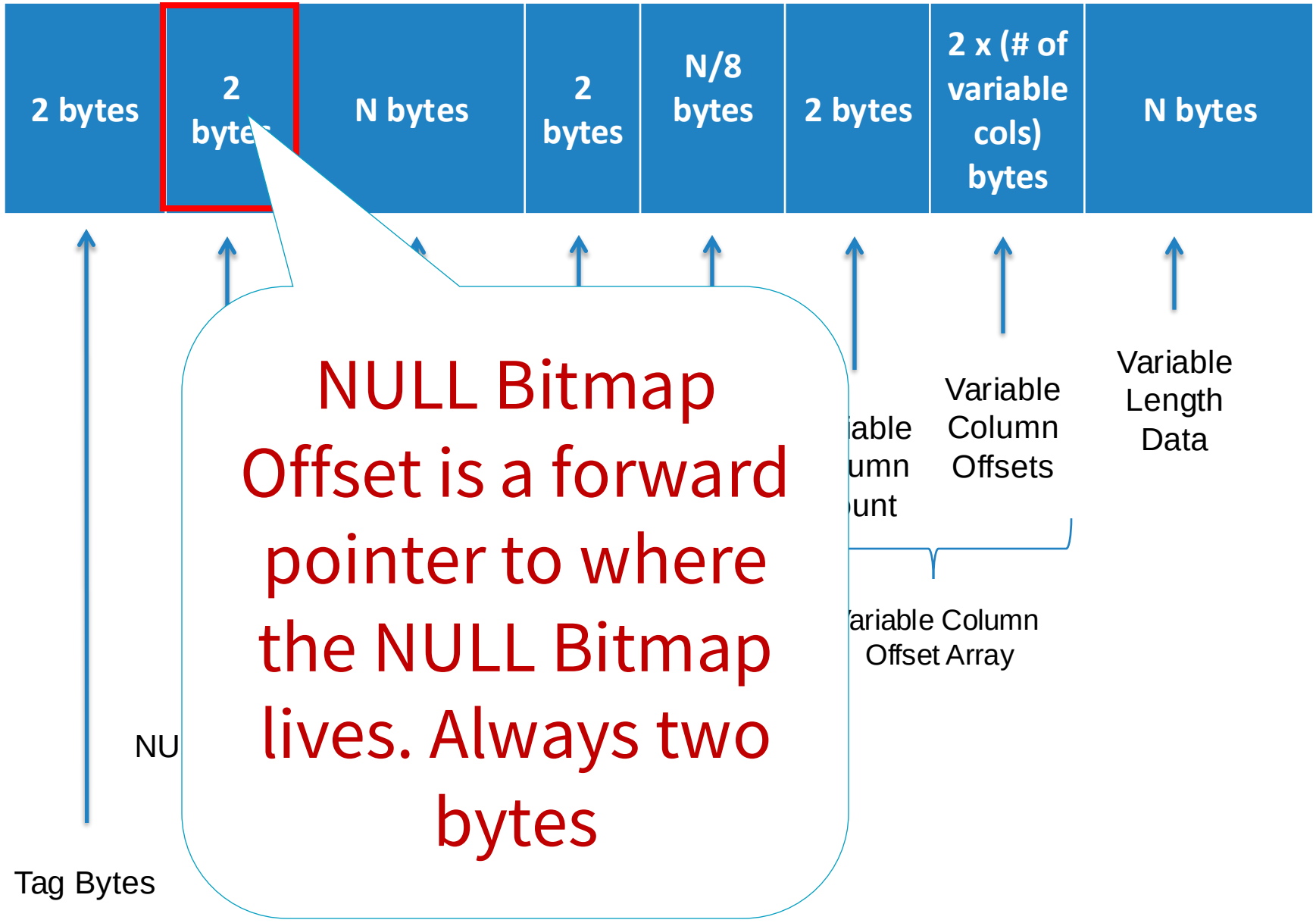
Row Structure



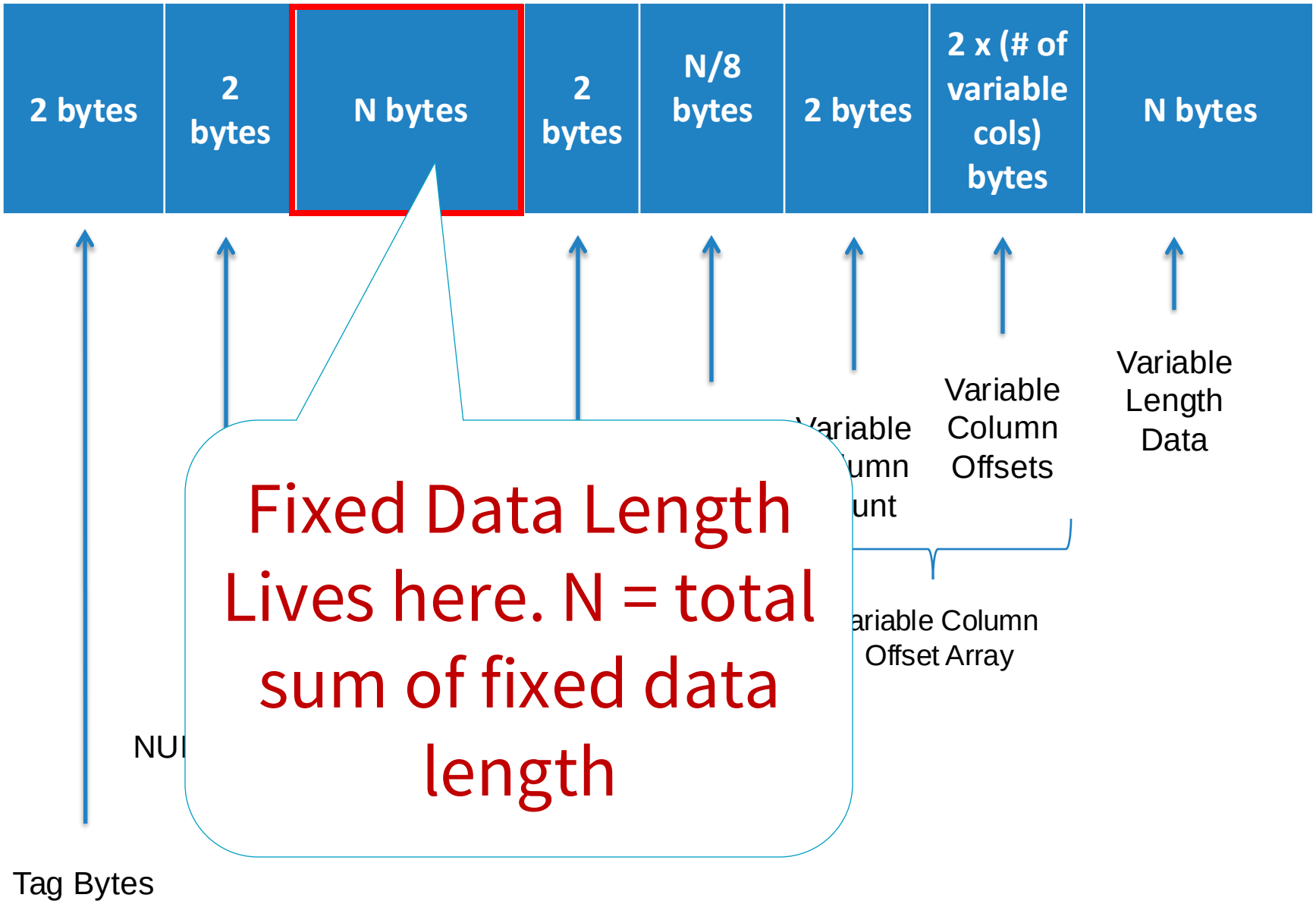
Row Structure



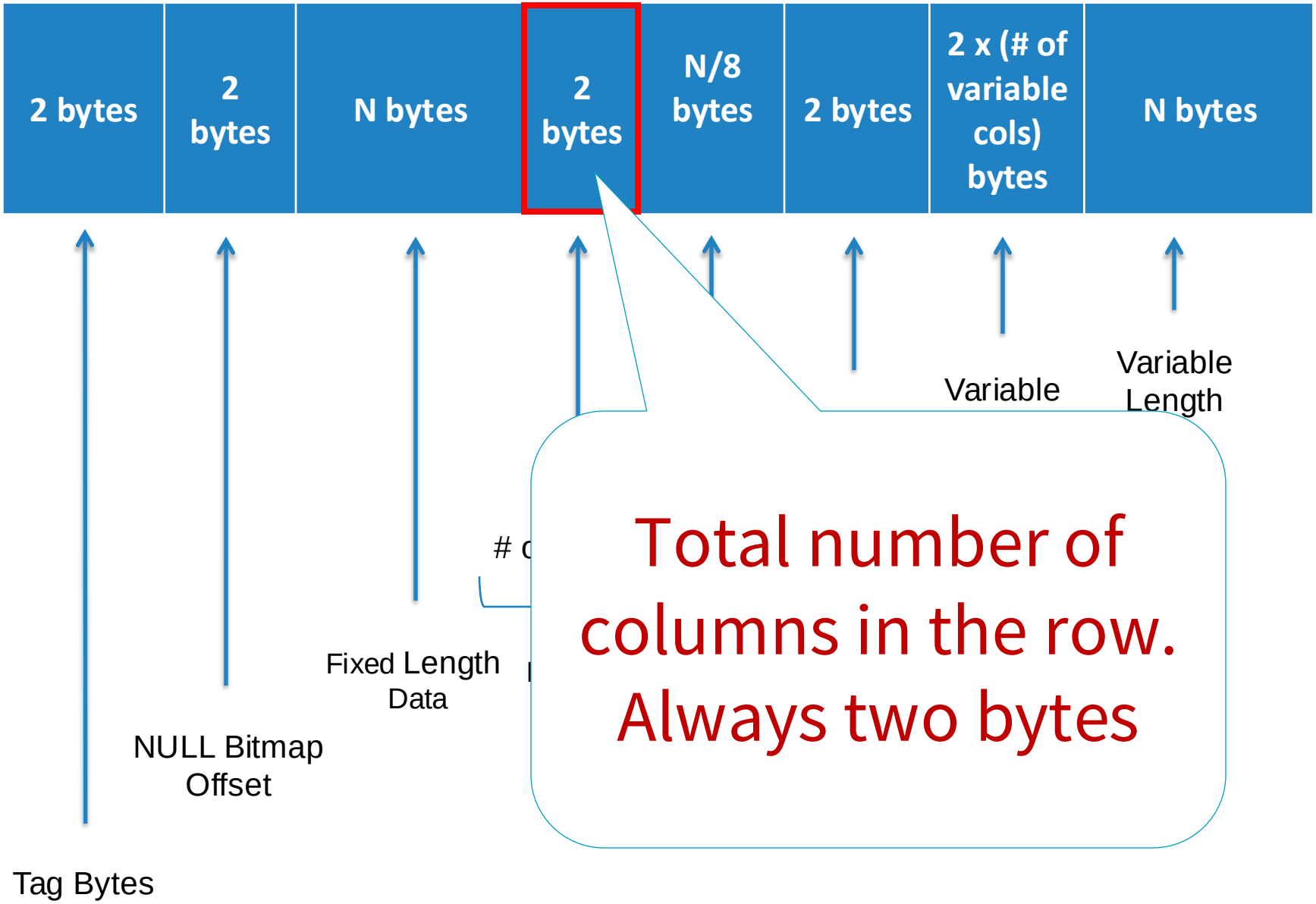
Row Structure



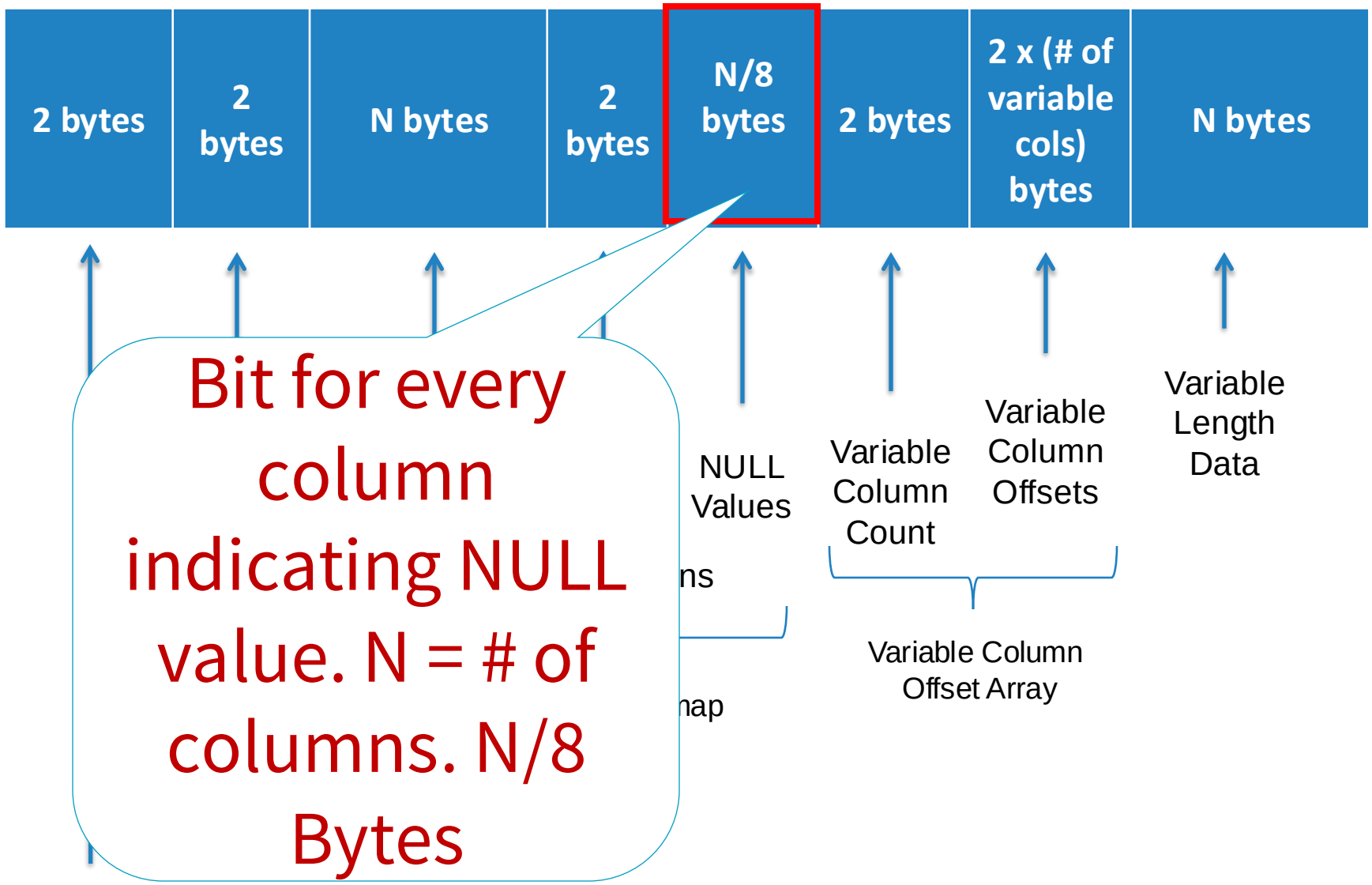
Row Structure



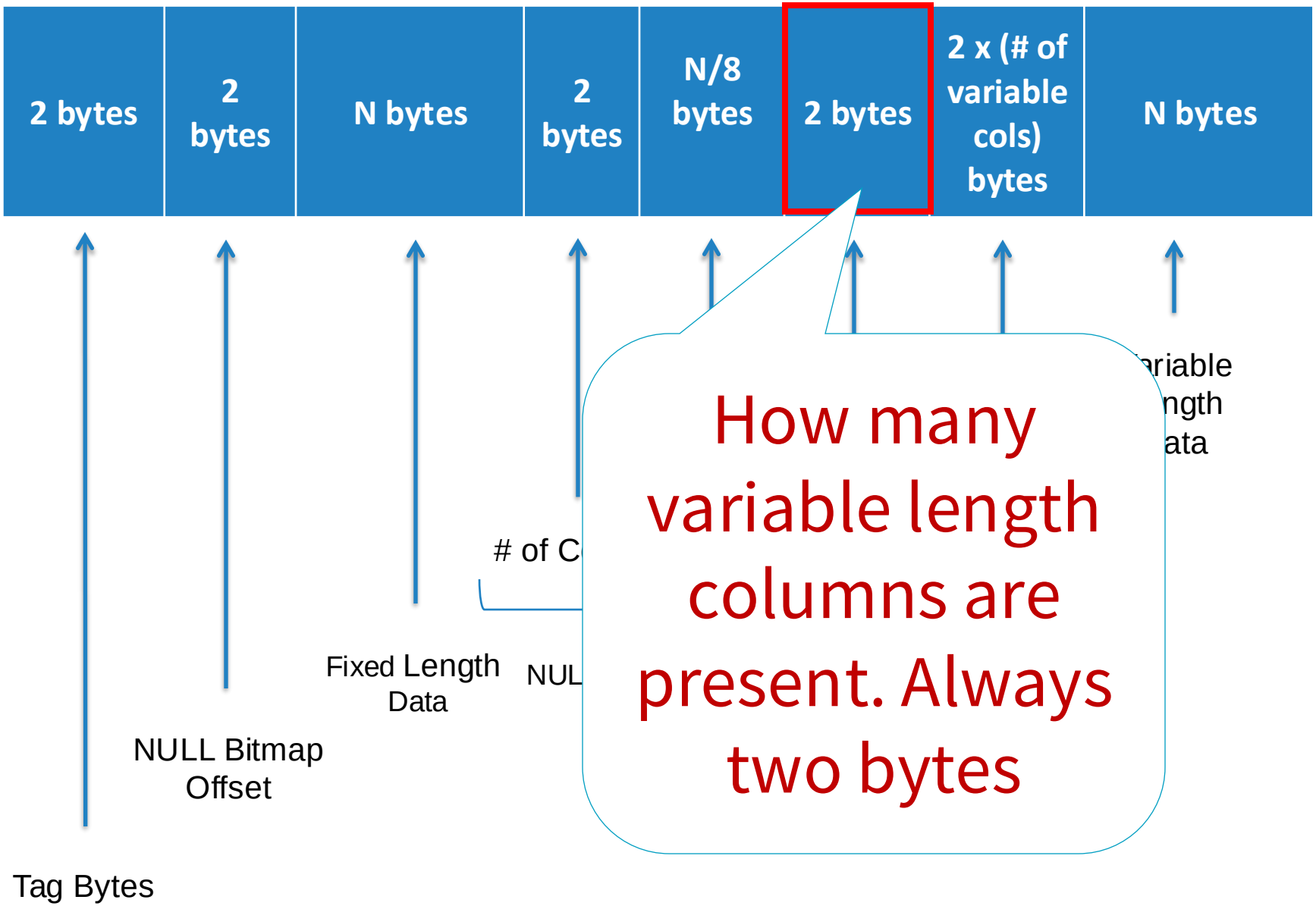
Row Structure



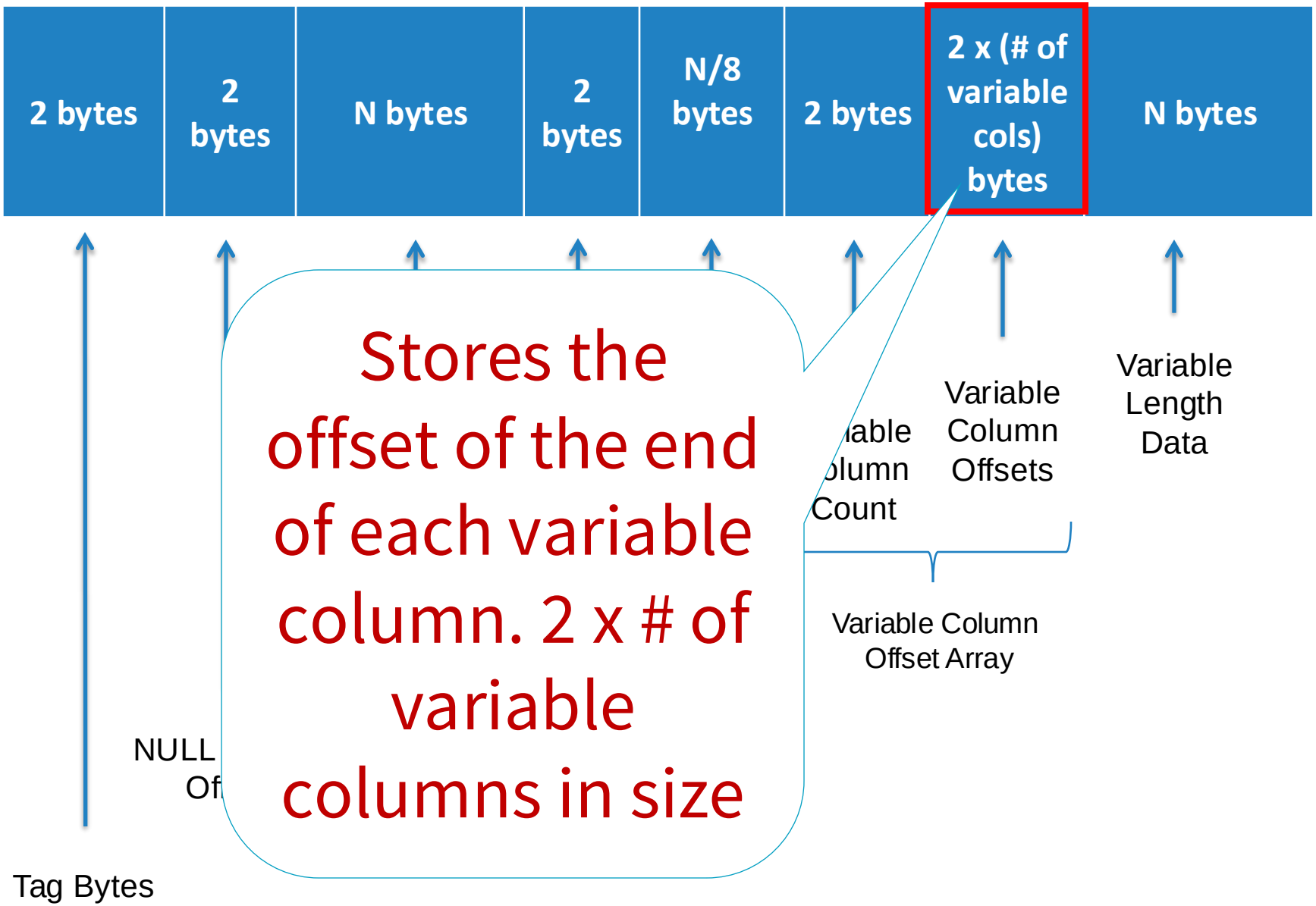
Row Structure



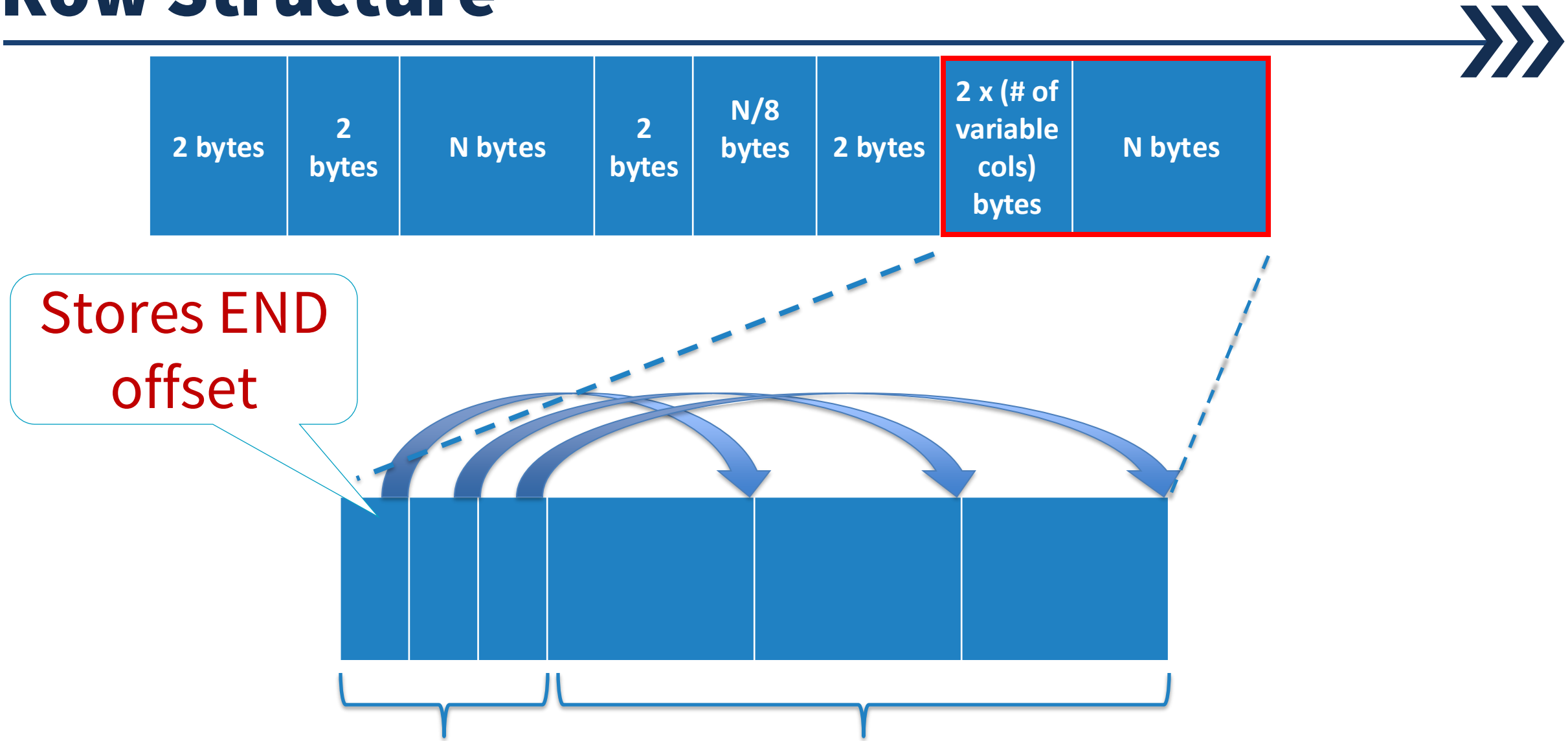
Row Structure



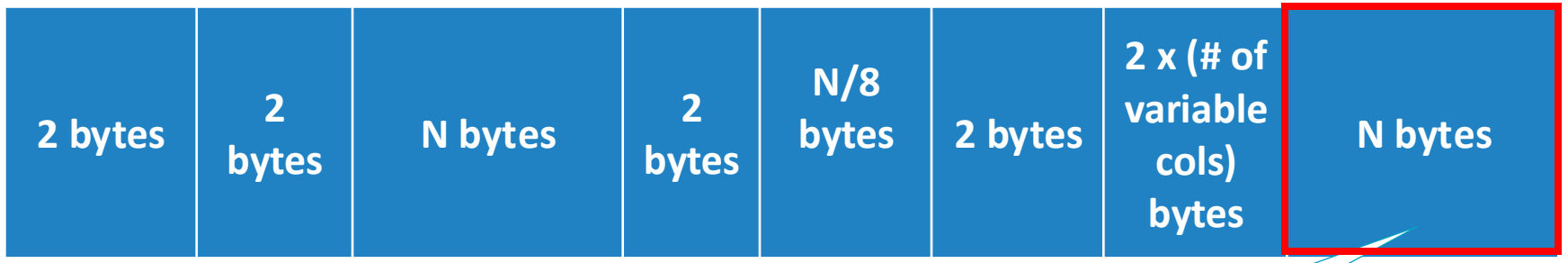
Row Structure



Row Structure



Row Structure



Variable column data lives here. N = total sum of variable data length.

Variable Length Data

Variable Column Offsets

Variable Column Count

Values

of Columns

NULL Bitmap

Variable Column Offset Array

How big is the row?

```
CREATE TABLE dbo.Customer(
        CustomerID INT
        , CustomerName VARCHAR(50)
        , CustomerState CHAR(2)
  );
```

Record Size

```
CREATE TABLE Customer(
    CustomerID INT,
    CustomerName
    VARCHAR(50),
    CustomerState
    CHAR(2)
  );
```

For the:	Bytes
Tag	?
NULL Offset Bitmap	?
Fixed Data Length	?
# of columns	?
Null Columns (N/8)	?
Variable Column Count	?
Variable Column Offset	?
Variable Length Columns	?
Total:	**?**

Record Size

```
CREATE TABLE Customer(
    CustomerID INT,
    CustomerName
    VARCHAR(50),
    CustomerState
    CHAR(2)
);
```

For the:	Bytes
Tag	2
NULL Offset Bitmap	2
Fixed Data Length	6
# of columns	2
Null Columns (N/8)	1
Variable Column Count	2
Variable Column Offset	2
Variable Length Columns	50
Total:	**67 bytes**

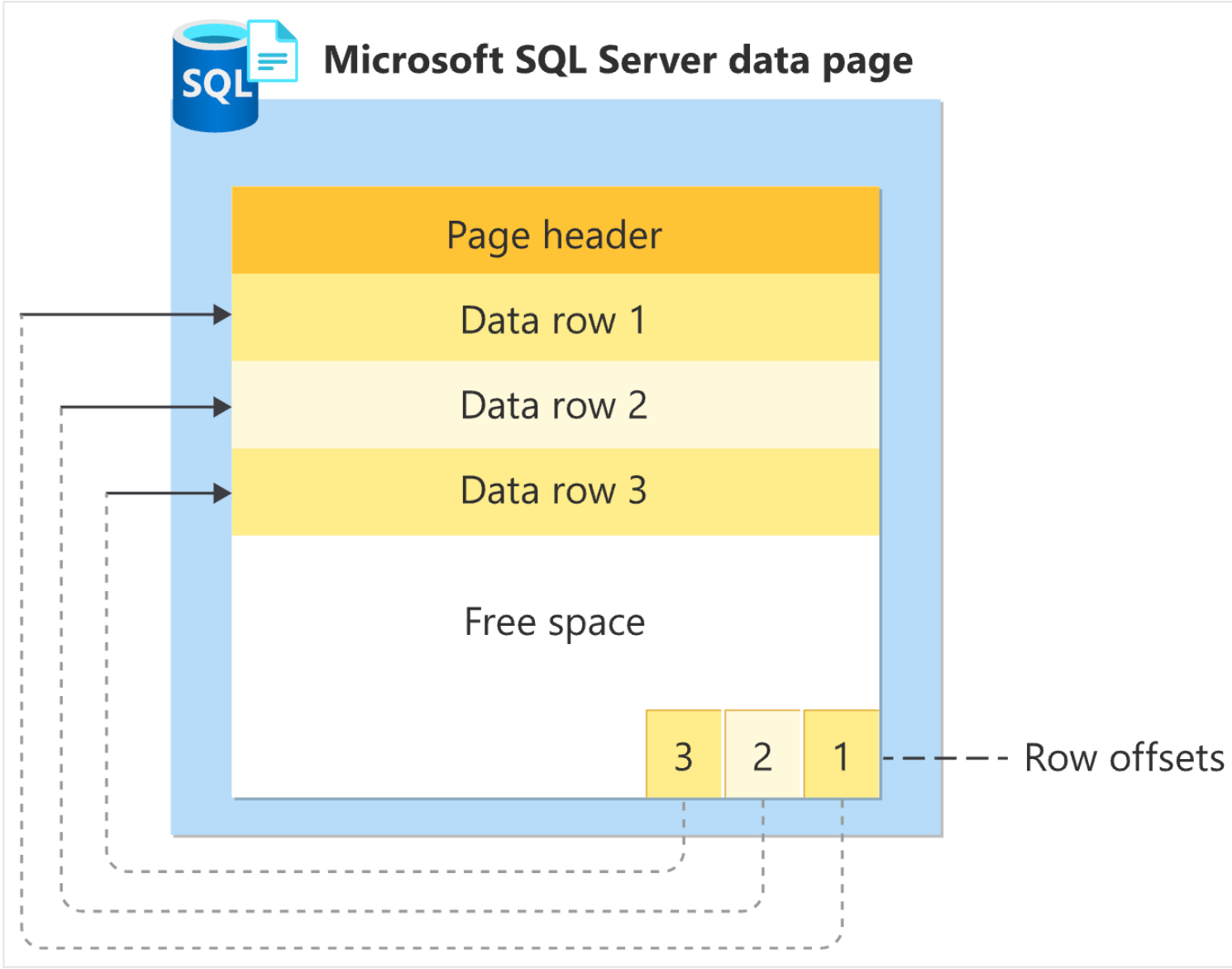
Pages

Pages

All pages are 8192 bytes (8k) in size.

Every page has a Header.

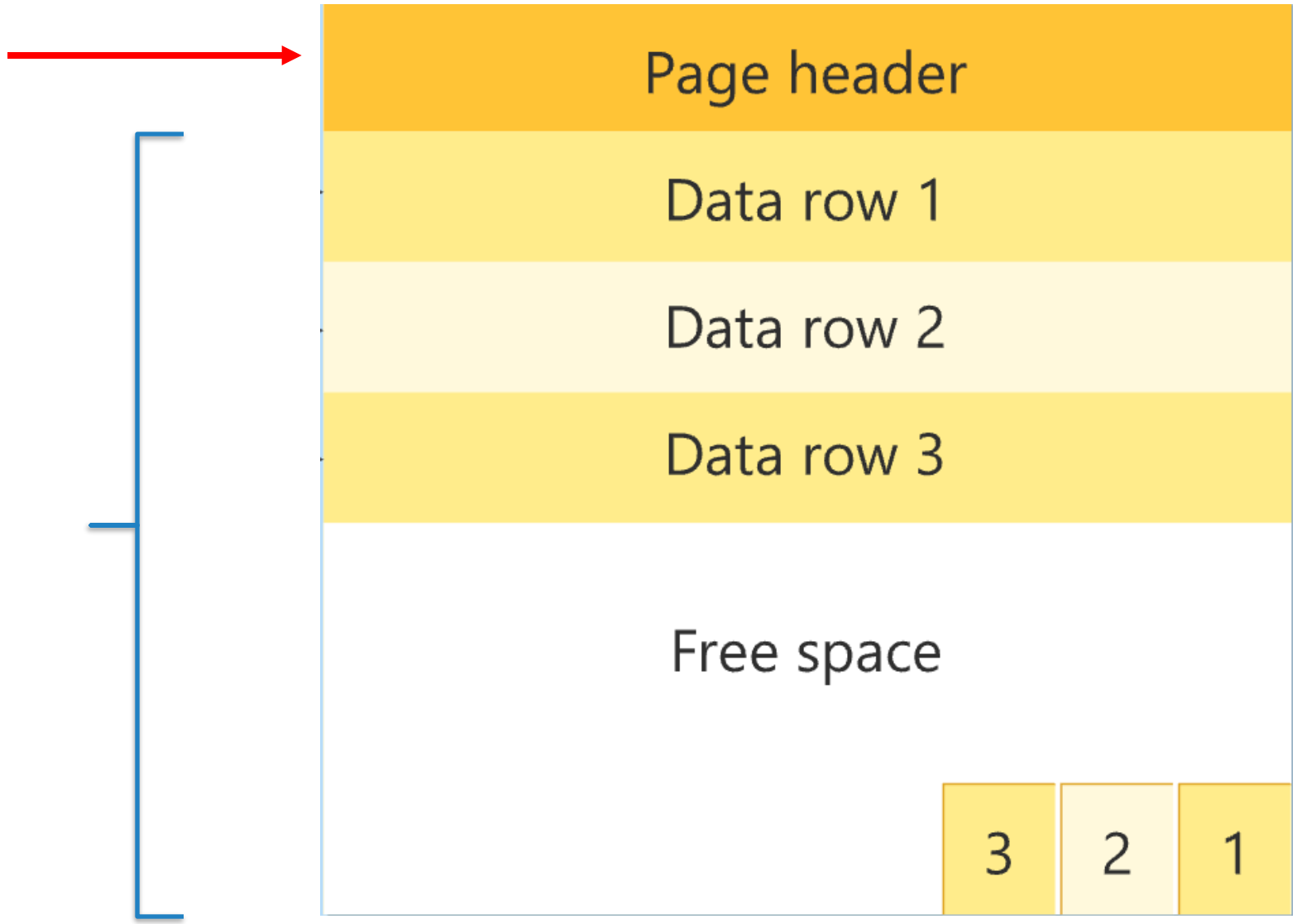
Every page has a slot array.



Pages

96 Bytes for Header

8096 Bytes for Rows



Header

Things to Note:

- Page Type
- Object ID
- Ghost Count Record
- Slot Count

```
PAGE: (1:696)


BUFFER:


BUF @0x000000000BFB02C0

bpage = 0x000000000B2B2000          bhash = 0x0000000000000000
bdbid = 6                           breferences = 0
bsampleCount = 1                    bUse1 = 4701
blog = 0x32159                      bnext = 0x0000000000000000

PAGE HEADER:


Page @0x000000000B2B2000

m_pageId = (1:696)                  m_headerVersion = 1
m_typeFlagBits = 0x0                m_level = 0
m_objId (AllocUnitId.idObj) = 147   m_indexId (AllocUnitId.idInd) =
Metadata: AllocUnitId = 72057594047561728
Metadata: PartitionId = 72057594045333504
Metadata: ObjectId = 1509580416     m_prevPage = (0:0)
pminlen = 90                        m_slotCnt = 2
m_freeData = 8182                   m_reservedCnt = 0
m_xactReserved = 0                  m_xdesId = (0:0)
m_tornBits = -627050546
```

```
PAGE: (1:696)


BUFFER:


BUF @0x000000000BFB02C0

bpage = 0x000000000B2B2000          bhash = 0x0000000000000000          bpageno = (1:696)
bdbid = 6                           breferences = 0                     bcputicks = 376
bsampleCount = 1                    bUse1 = 4701                        bstat = 0xc00009
blog = 0x32159                      bnext = 0x0000000000000000

PAGE HEADER:


Page @0x000000000B2B2000

m_pageId = (1:696)                  m_headerVersion = 1                 m_type = 10
m_typeFlagBits = 0x0                m_level = 0                         m_flagBits = 0x200
m_objId (AllocUnitId.idObj) = 147   m_indexId (AllocUnitId.idInd) = 256
Metadata: AllocUnitId = 72057594047561728
Metadata: PartitionId = 72057594045333504                               Metadata: IndexId = 1
Metadata: ObjectId = 1509580416     m_prevPage = (0:0)                  m_nextPage = (0:0)
pminlen = 90                        m_slotCnt = 2                       m_freeCnt = 6
m_freeData = 8182                   m_reservedCnt = 0                   m_lsn = (41:3993:669)
m_xactReserved = 0                  m_xdesId = (0:0)                    m_ghostRecCnt = 0
m_tornBits = -637050546
```

Pages

Single page size limit for data is 8060 bytes	Slot Array	The rows do not have to be stored in order physically on the page
No exceptions Can store single record or multiple records Off row storage	Stores the offsets to each row on the page 2 bytes per row	The slot array offsets will be stored in sorted order

Pages

Records are stored on the same type of pages

IE: Data records are stored on Data pages
IE: Index records are stored on Index pages

Boot page

Page (1:9) (file:page#)
Store metadata about the database
- Very critical page.
- If corrupt, restore is the only option

Internal Pages

PFS – Page Free Space

SGAM – Shared Global Allocation Map

GAM – Global Allocation Map

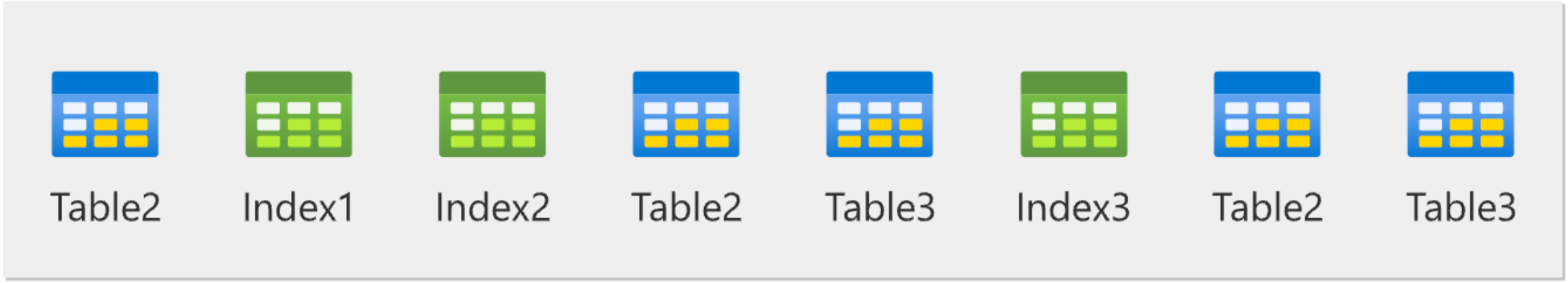
IAM – Index Allocation Map

Internal Pages

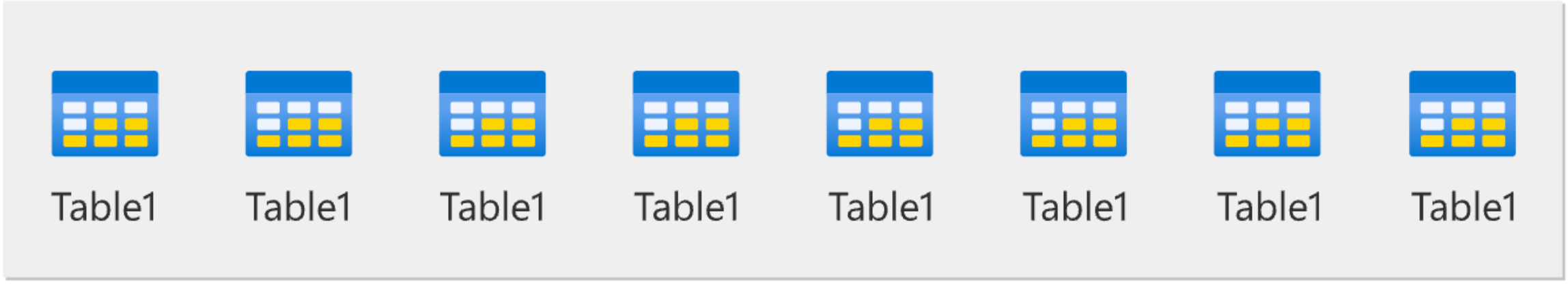
Type	Purpose	Size/Range
PFS	Tracks page allocation and page free space	1 PFS for every 8088 pages
SGAM	Tracks used mixed extents	1 SGAM for every 64,000 extents (4GB)
GAM	Tracks used uniform extents	1 GAM for every 64,000 extents (4GB)
IAM	Maps extents in a 4-GB part of file used by an allocation unit	4GB

Extents

Mixed extent



Uniform extent



Tools

DBCC IND

DBCC PAGE

Sys.fn_PhysLocFormatter

Undocumented
(Sort of)
Unsupported
(Sort of)

Sys.dm_db_database_page_allocations

sys.dm_db_page_info

Not Supported by Microsoft. There are a TON of people online willing to help with issues and/or questions.

DBCC IND

DBCC IND(

<DatabaseName>,<‘tablename’>, <Index_ID>

);

For example:

DBCC IND(AdventureWorks,’Person.Person’,1)

DBCC PAGE

DBCC PAGE(

<database_name>, <fileid>, <pagenumber>,<detail_level>

);

Note: Must execute DBCC TRACEON(3604) BEFORE DBCC PAGE!

0 = Header
1 = Header/hex dump for rows
2 = Header/page dump
3 = Header/detail row info

For example:

DBCC PAGE(AdventureWorks, 1, 696, 3)

Dynamic Management Functions

```
SELECT * FROM
sys.dm_db_database_page_allocation(
DatabaseId , TableID , IndexID,
PartitionID , Mode )
```

```
SELECT * FROM sys.dm_db_page_info (
DatabaseId , FileId , PageId , Mode )
```


Demo


***STOP: 0x000000D1 (0x00000000, 0xF73120AE, 0xC0000008, 0xC0000000)
A problem has been detected and Windows has been shut down to prevent damage to your computer
DRIVER_IRQL_NOT_LESS_OR_EQUAL
If this is the first time you've seen this Stop error screen, restart your computer. If this screen appears again, follow these steps:
Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any windows updates you might need.
If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.
**** ABCD.SYS - Address F73120AE base at C0000000, DateStamp 36B072A3
Kernel1 Debugger Using: COM2 (Port 0x2F8, Baud Rate 19200)
Beginning dump of physical memory
Physical memory dump complete. Contact your system administrator or technical support group.


JUST KIDDING!

How many rows?

```
CREATE TABLE dbo.Sales(
    CustomerID INT,
    CustomerName
    VARCHAR(50),
    CustomerState
    CHAR(2)
  );
```

For the:	Bytes
Tag	2
NULL Offset Bitmap	2
Fixed Data Length	6
# of columns	2
Null Columns (N/8)	1
Variable Column Count	2
Variable Column Offset	2
Variable Length Columns	50
Total:	**67 bytes**

How many rows?

```
CREATE TABLE
dbo.Sales(
    CustomerID INT,
    CustomerName
       VARCHAR(50),
    CustomerState
       CHAR(2)
  );
```

8060 / 69 bytes = 116 rows per page

Summary

Row Density	Page Density
How much data do we have in the row?	How many rows do we have on the page?

Summary

Understanding the internals is critical for everything.

Smart table design will lead to:

More rows on a page

- Less Pages to store the data

- - Less Work for SQL Server to read the pages

- - - Faster response times

BREAK

15:00

Transaction Log File

Consider

```
UPDATE dbo.Customer

SET OfficePhone =
'4025559710'

WHERE CustomerName =
'WideWorldImporters'
```

Transaction Log

SQL Server locates the 8k page with that record

Reads Pages into buffer pool (memory)

Updates in-memory pages & generates log records

Writes the page to the transaction log

Marks the transaction as complete

Transaction Log

Single or multiple log files

Tracks all transactions including any database modifications

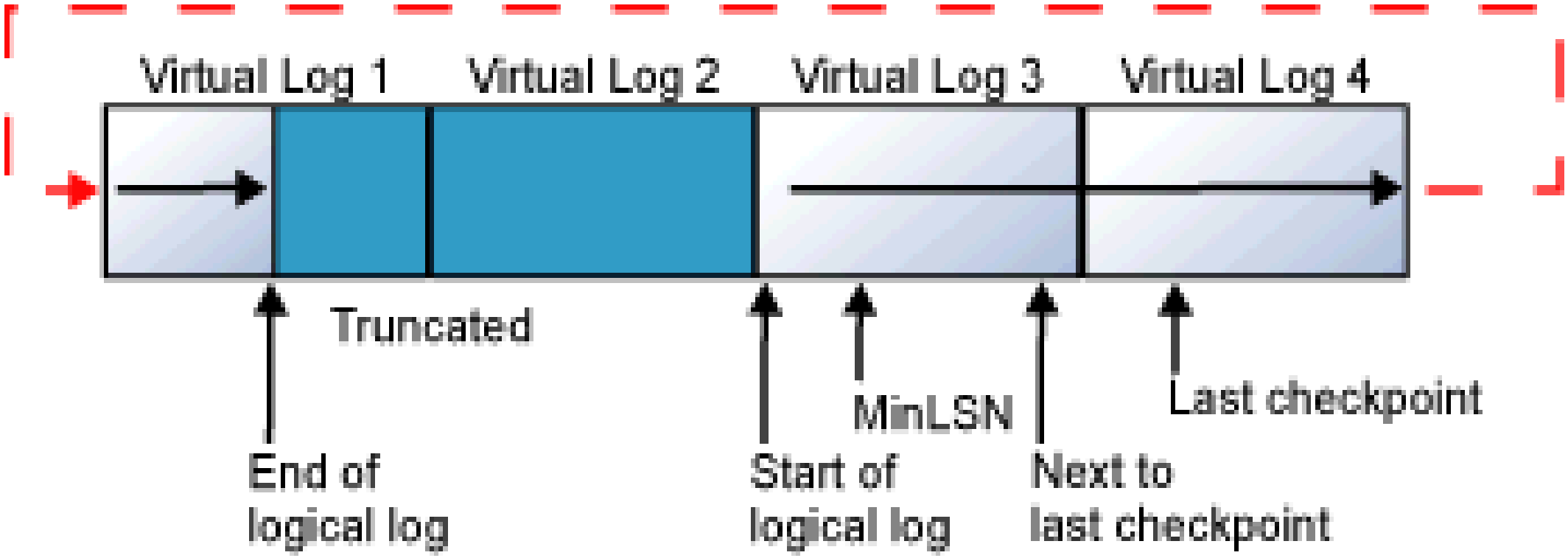
Facilitates point-in-time-restores (PITR) and any rollbacks

Sequential in nature – LSN [VLF ID:Log Block ID:Log Record ID]

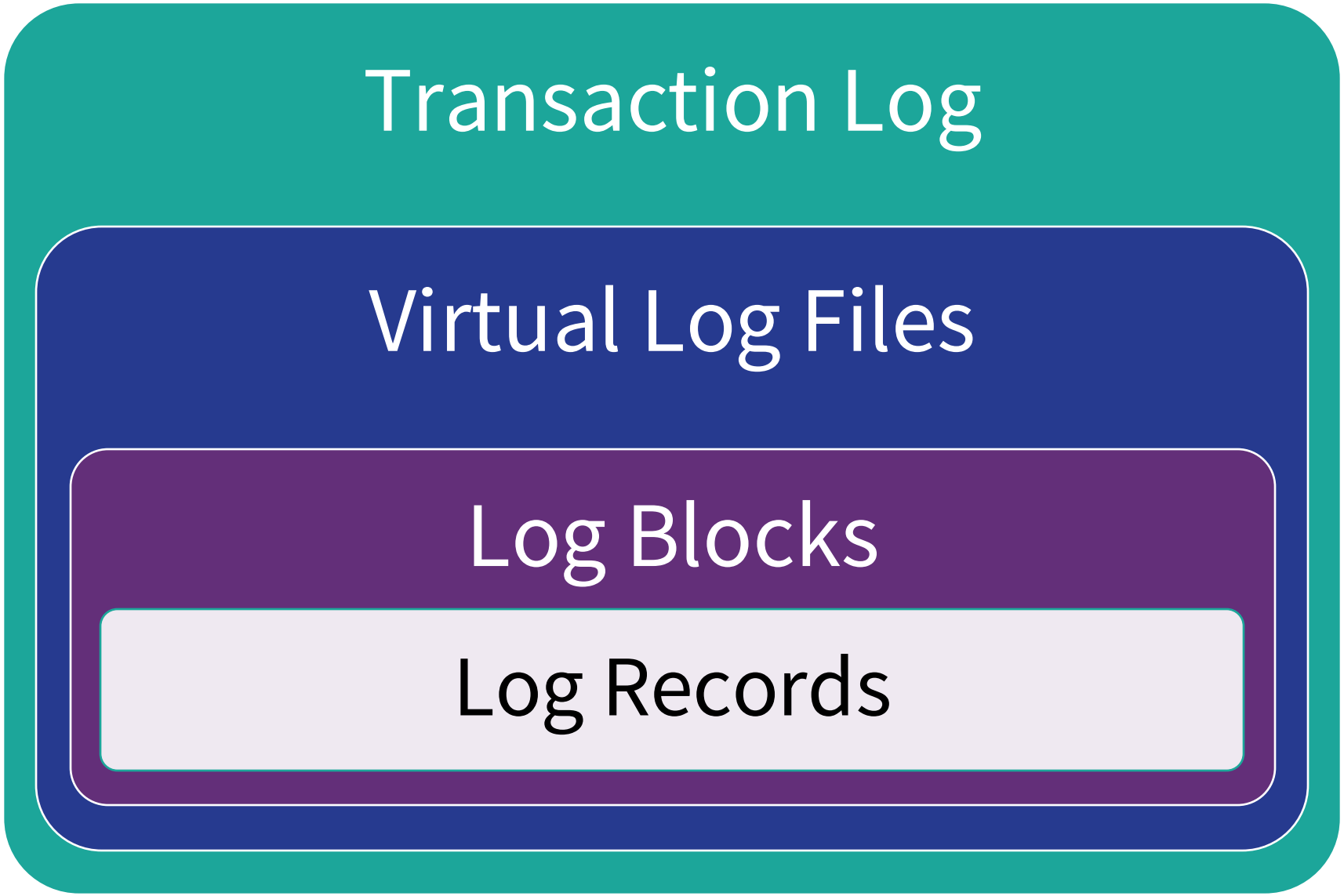
Accelerated Database Recovery drastically changes how the transaction log operates.

Transaction Log Files

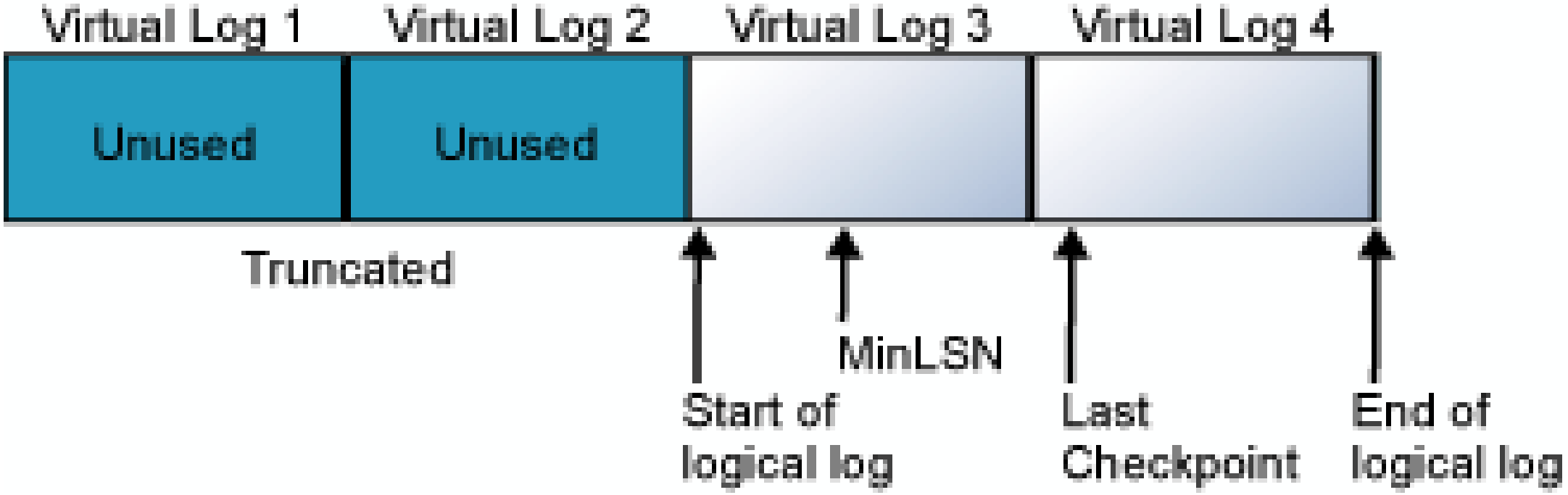
The transaction log is a wrap-around file



Transaction Log Files



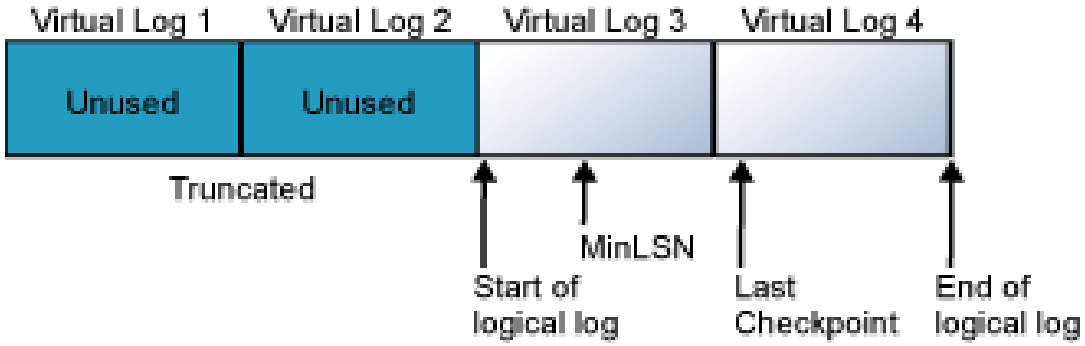
Virtual Log Files



Virtual Log Files

VLFs have no fixed size

No fixed number of VLFS



The 2 above items are not adjustable

Size is dependent on transaction log file growth settings

Log Blocks

A log block is the basic unit of I/O for transaction logging.

Log blocks contain log records that is the basic unit of transaction logging when writing log records to disk.

Each VLF contains one or more log blocks

Each block varies in size & is always an integer multiple of 512 bytes. Maximum size is 60 kilobytes

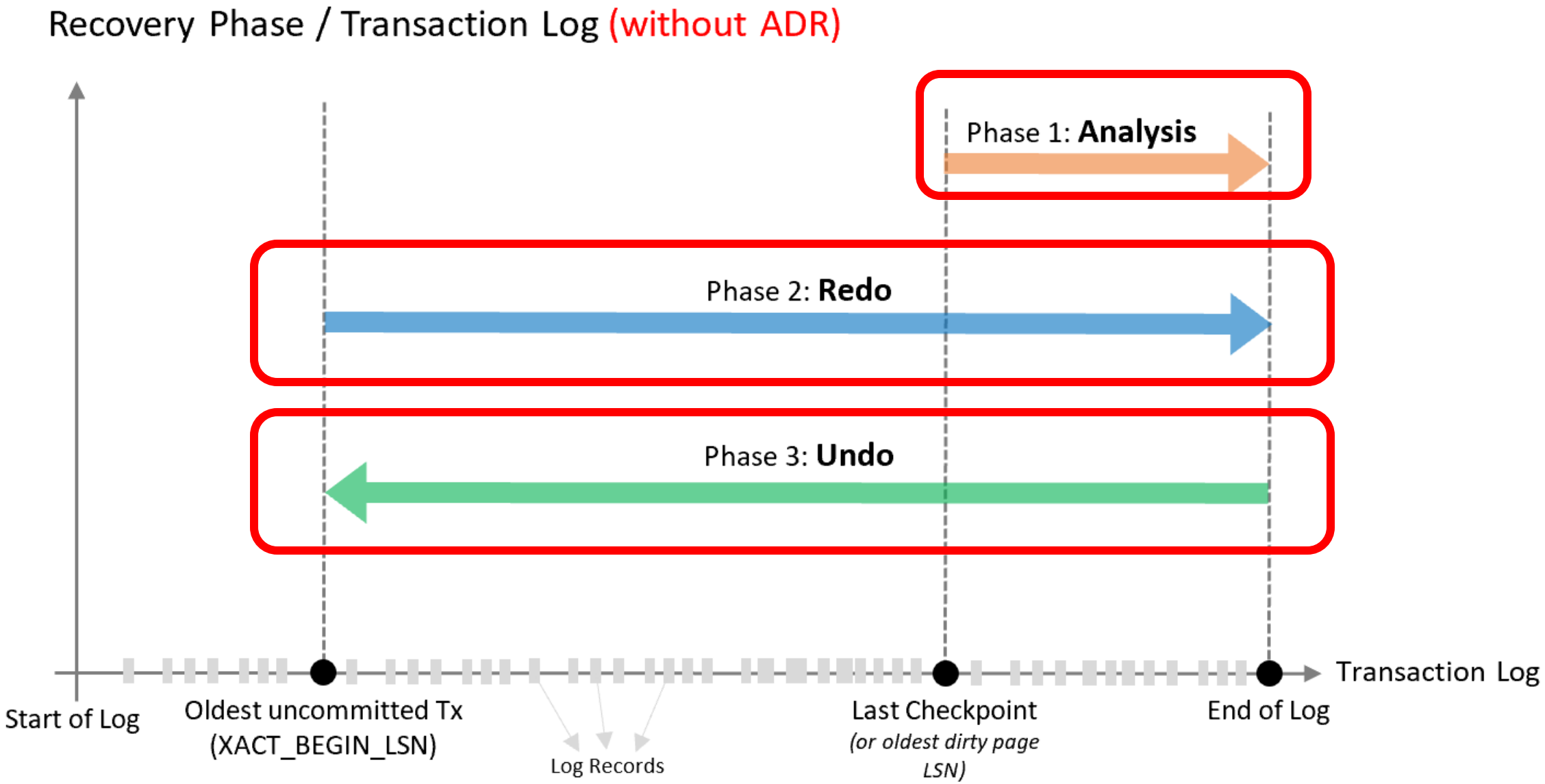
Log Blocks

Each log block is uniquely addressed by its block offset inside the virtual log files.

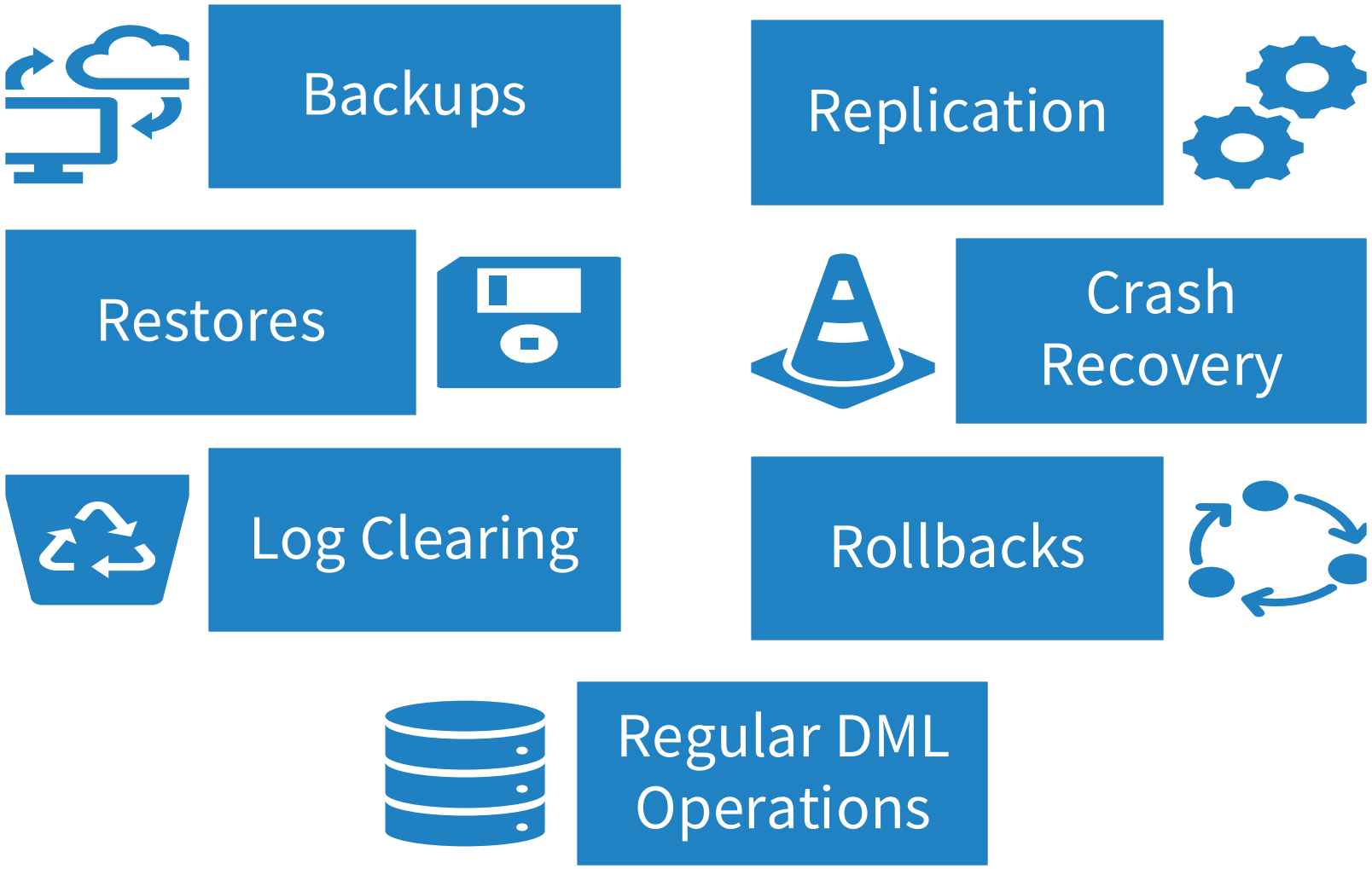
The first block always has a block offset that points past the first 8 KB in the VLF.

If a log block will not fit inside the VLF, an empty log block is created and the previous log block is written to the subsequent VLF

Transaction Log Files



VLF PERFORMANCE PROBLEMS



VIRTUAL LOG FILES

A large number of VLFs can slow things down.

VLF counts under 300 ideally

Performance boost on startup, insert/update/delete & backup/restore operations, also affects AG failovers

HOW TO GET THE COUNTS

VIRTUAL LOG FILES

DBCC LOG

SQL 2016 SP 2

Return all LSNs

TOTAL VLFs is COUNT of those rows

```
SELECT name AS 'Database Name',
total_vlf_count AS 'VLF count'
FROM sys.databases AS s
CROSS APPLY sys.dm_db_log_stats(s.database_id)
--WHERE total_vlf_count > 100;
```

WHAT CAUSES HIGH VLFs?

Inappropriate log file sizing

Auto-growth settings

Each growth event adds VLFs to the log file

Small Growth Segments = High VLF Counts

FIXING VIRTUAL LOG FILES

Issue a CHECKPOINT

Backup the Transaction Log

Shrink Log File to Smaller Size (Truncate Only)

Regrow Log File in chunks back to the current size

WHAT CHUNKS TO GROW IN?

GROWTH SIZE	TOTAL VLFS CREATED
<64MB	4
>64 and <=1GB	8
>1GB	16

Optimal to grow in

8000MB Chunks

to Minimize Amount of VLFs

Great Resource by Kimberly Tripp
https://www.sqlskills.com/blogs/kimberly/transaction-log-vlfs-too-many-or-too-few/

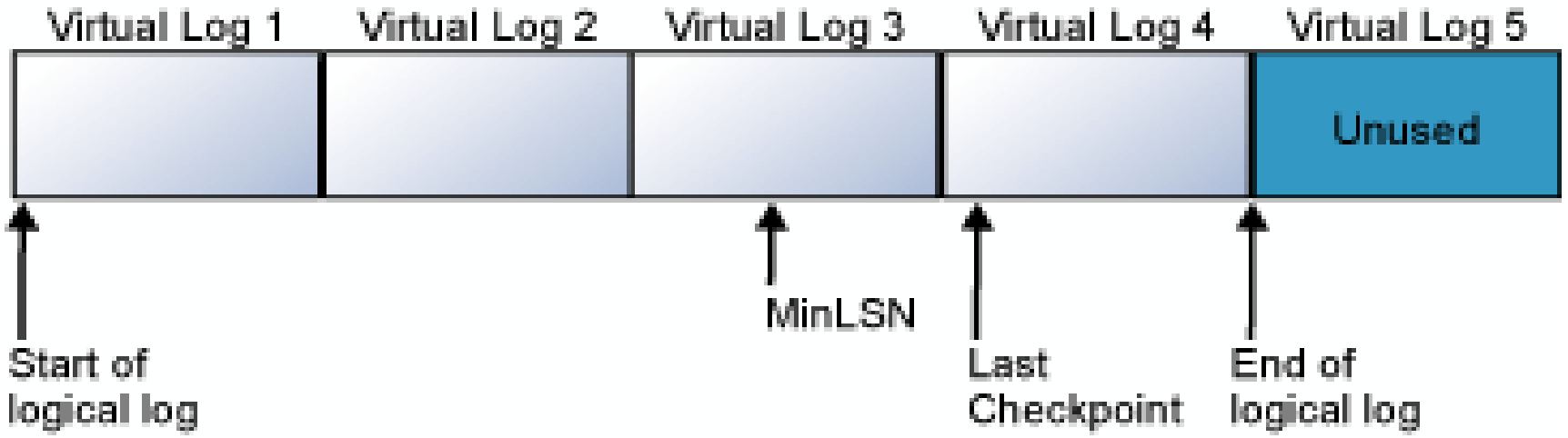
Log Truncation

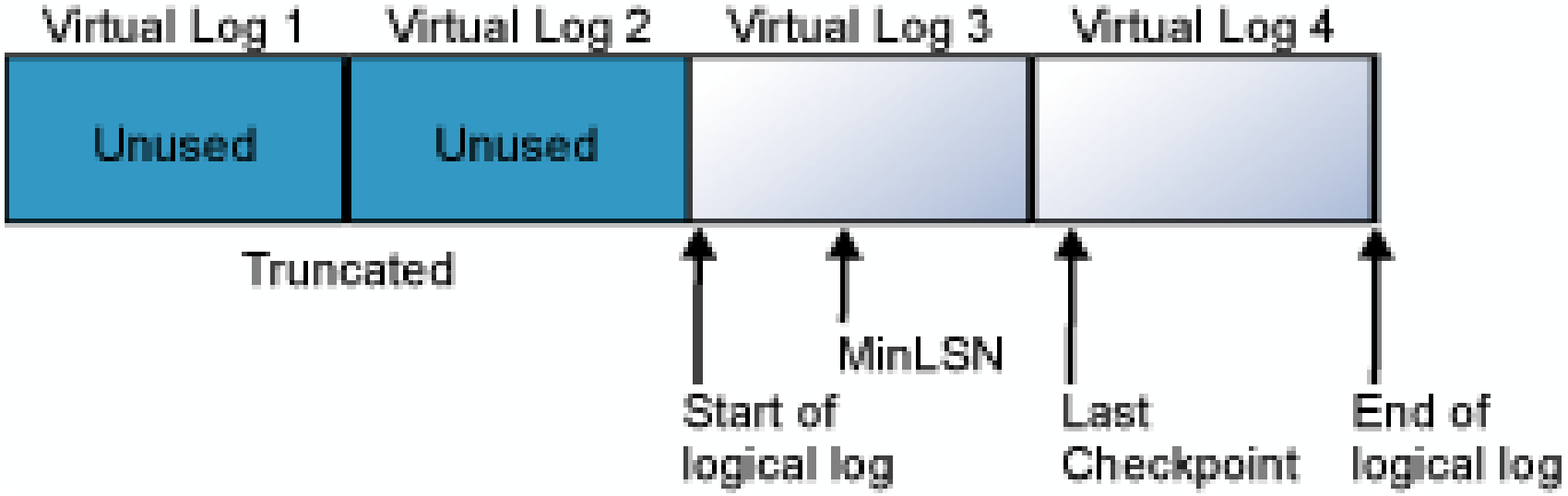
Deletes inactive virtual log files from the logical transaction log, freeing space for log reuse

Checkpoint must occur prior to log truncation

Delayed Log Truncation Reasons	
Active Transaction	Change Data Capture (CDC)
Replication	Active Backup/Restore
Availability Group Connectivity	Log Backups
Database Mirroring (Deprecated)	Checkpoint

Log Truncation





Available Tools

fn_db_log()

```
SELECT * FROM fn_dblog
(
NULL, -- Start LSN nvarchar(25)
NULL-- End LSN nvarchar(25)
)
```

fn_full_dblog()

```
SELECT * FROM sys.fn_full_dblog
 (NULL, -- Start LSN nvarchar (25)
  NULL, -- End LSN nvarchar (25)
  NULL, -- DB ID int
  NULL, -- Page file ID int
  NULL, -- Page ID int
  NULL, -- Logical DB ID nvarchar (260)
  NULL, -- Backup Account nvarchar (260)
  NULL -- Backup Container nvarchar (260)
 )
```

Transaction Log Operations

OPERATION	DESCRIPTION
LOP_ABORT_XACT	Indicates that a transaction was aborted and rolled back.
LOP_BEGIN_CKPT	A checkpoint has begun.
LOP_BEGIN_XACT	Indicates the start of a transaction.
LOP_COMMIT_XACT	Indicates that a transaction has committed.
LOP_CREATE_INDEX	Creating an index.
LOP_DELETE_ROWS	Rows were deleted from a table.
LOP_DELETE_SPLIT	A page split has occurred. Rows have moved physically.
LOP_DROP_INDEX	Dropping an index.
LOP_END_CKPT	Checkpoint has finished.


Demo

What does this mean?

If we know that -

The transaction log is sequential

It records everything transactional including database modifications

Everything gets written to the log first

We should then put the transaction log file on the…..

What does this mean?

FASTEST

STORAGE

POSSIBLE

Further more

Appropriate growth settings matter

Instant File Initialization doesn’t apply*

Potentially arguably more critical than the data file

Resources

DBCC IND/PAGE Paul Randal - http://www.sqlskills.com/blogs/paul/category/dbcc/

Data Types - http://msdn.microsoft.com/en-us/library/ms187752.aspx

SGAM, GAM, IAM, PFS, DIFFMAP http://blogs.msdn.com/b/sqlserverstorageengine/archive/2006/07/08/under-the-covers-gam-sgam-and-pfs-pages.aspx

Anatomy of a Record (Paul Randa) http://www.sqlskills.com/blogs/paul/inside-the-storage-engine-anatomy-of-a-record/

Questions?
Answers!

Your feedback is important to us

Evaluate this session at:

www.PASSDataCommunitySummit.com/evaluation

Thank you

Message for the end of the presentation goes here

John Morehouse

@SQLRUS

https://sqlrus.com

https://linkedin.com/in/johnmorehouse

Slides & Demos