

Resurrecting Lost Data: Mastering SQL Server's Transaction Logs and Row Internals Secrets



by Vladimir Afanasev

Data Loss happens 💩

Why do we need Transaction Log?



Chronological Record

Documents all database modifications in sequence. Captures every change made



Essential for Recovery

Enables point-in-time restoration. Critical for business continuity planning



ACID Guarantees

Ensures Atomicity, Consistency, Isolation, Durability. Fundamental to database reliability

Transaction ID	Timestamp	AllocUnitName	Operation	LSN
0000:00000452	2025-05-21 16:22:15.123	1798297466	BEGIN XACT	7314
0000:00000452	2025-05-21 16:22:18.456	1798297466	LOP_DELETE_ROWS	7315
0000:00000452	2025-05-21 16:22:19.456	1798297466	COMMIT XACT	7316

Transaction Log details

Log Sequence Numbers (LSNs)

Unique identifiers for log records.
Enable precise tracking of database
changes



Transaction context

Each data modification is executed in
the context of transaction

Stores data as Binary values

Less space, unified and precise format

Data Row Storage secrets



Data Pages

SQL Server stores data in 8KB pages



Row Header

Contains metadata about the row



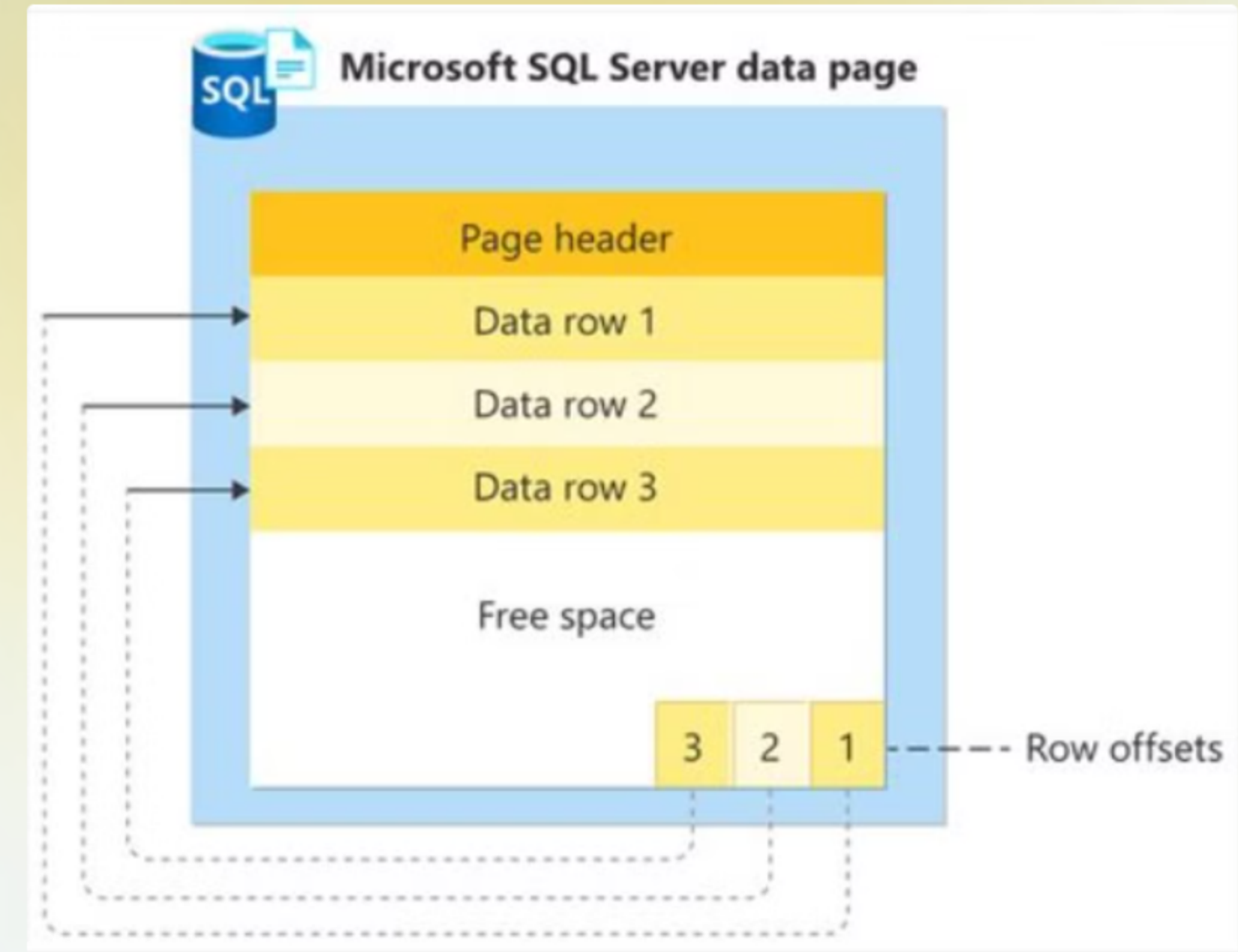
Data Rows

Stored as binary values



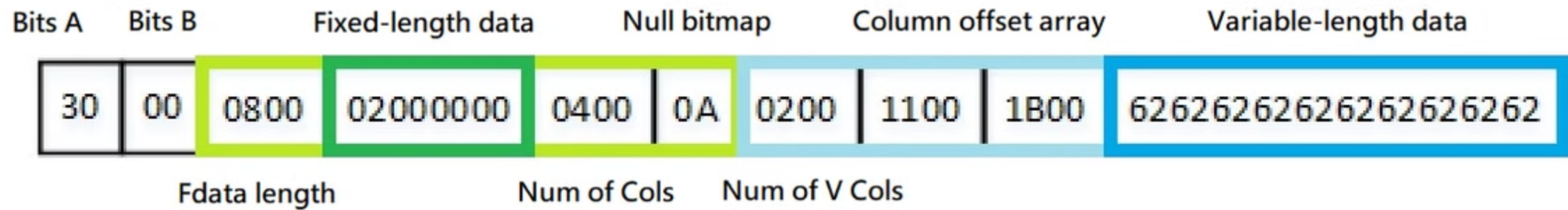
Row offsets

Indicates the logical order of Data rows on the page



Fixed-length vs Variable-length Data types

Feature	CHAR	VARCHAR
Definition	Fixed-length character data type	Variable-length character data type
Storage Behavior	Always takes up the full defined length (e.g. CHAR(10) always uses 10 bytes)	Uses only the needed space +2 bytes for length storage
Performance	Faster for fixed-size data because it doesn't require length calculation	More efficient for variable-length data, saving space but slightly increasing processing time
Usage Scenario	Best for storing fixed-length strings like Country codes ('US','UK'), Currency codes, etc	Best for storing variable-length text like names, descriptions or comments



What is inside Data Row?

```
CREATE TABLE dbo.Fixed  
(  
Col1 char(5) NOT NULL,  
Col2 int NOT NULL,  
Col3 char(3) NULL,  
Col4 char(6) NOT NULL  
);
```

--INSERT just one row for the Demo

```
INSERT dbo.Fixed VALUES ('ABCDE', 123, NULL, 'CCCC');
```

--DELETE without WHERE

```
DELETE FROM dbo.Fixed
```


Recovering a Deleted Row

1. Identify DELETE operation

Find the LOP_DELETE_ROWS record. Filter by table name and approximate time:

```
SELECT [ObjectId], [TransactionId], [RowLog Contents 0]
FROM fn_dblog(NULL, NULL)
WHERE [AllocUnitName] LIKE '%Customers%'
AND [Operation] = 'LOP_DELETE_ROWS'
```

2. Extract and parse row data


Interpret the RowLog Contents hex value. Get column names and Data types:

```
SELECT @FixedLengthSize = dbo.UDF_GetIntFromNtoMBytes([RowLog Contents])
```

3. Convert binary values back

Use extracted values to create INSERT statement. Restore the lost data:

```
CASE
  WHEN [SystemTypeId] IN (231, 239)
  THEN CONVERT(NVARCHAR(MAX), [HexValue])
  WHEN [SystemTypeId] = 48
  THEN CONVERT(TINYINT, [HexValue])
END
```

 EXEC [dbo].[ResurrectionOfLostData]
@SchemaTableName = 'dbo.Fixed'

	SlotId	ObjectId	PartitionId	TransactionId	AllocUnitId	TypeDesc	RowLog Contents 0
1	0	1798297466	72057594071613440	0000:0006a825	72057594093502464	IN_ROW_DATA	0x1000160041424344457B000000000000434343202004...

	SlotId	RowId	Name	DataTypeName	NullBit	SystemTypeId	BitPos	Precision	Scale	IsNull	ColumnValueStartPosition	ColumnLength	HexValue
1	0	1	Col1	char	1	175	0	0	0	0	5	5	0x4142434445
2	0	1	Col2	int	2	56	0	10	0	0	10	4	0x7B000000
3	0	1	Col3	char	3	175	0	0	0	1	14	3	NULL
4	0	1	Col4	char	4	175	0	0	0	0	17	6	0x434343432020

	RowId	Col1	Col2	Col3	Col4
1	1	ABCDE	123	NULL	CCCC

Key points to remember tomorrow



Use BEGIN TRAN before DELETE

Always allow you to ROLLBACK

Aa

Data types matter

Know your data and choose the proper one



NB: TRUNCATE is not recoverable

It's minimally logged

Wanna learn more?



Stored procedure on GitHub

<https://github.com/NotYetBenGan/base/blob/main/SQL/ResurrectionOfLostData.sql>



Row Internals

<https://www.sqlservercentral.com/articles/understanding-the-internals-of-a-data-page>



Transaction log

<https://www.sqlservercentral.com/articles/reading-sql-servers-transaction-log>

